

AD/A-004 159

COMPUTER RECOGNITION OF FACIAL PROFILES

Gerald J. Kaufman

Ohio State University

Prepared for:

Air Force Office of Scientific Research

August 1974

DISTRIBUTED BY:

NTIS

National Technical Information Service
U. S. DEPARTMENT OF COMMERCE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER AFOSR - TR - 75 - 0074	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER AD/A-004159	
4. TITLE (and Subtitle) COMPUTER RECOGNITION OF FACIAL PROFILES		5. TYPE OF REPORT & PERIOD COVERED Interim	
		6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s) Gerald J. Kaufman		8. CONTRACT OR GRANT NUMBER(s) AFOSR 71-2043	
9. PERFORMING ORGANIZATION NAME AND ADDRESS The Ohio State University Department of Electrical Engineering Columbus, Ohio 43210		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 9769-02	
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research (NM) 1400 Wilson Blvd Arlington, Virginia 22209		12. REPORT DATE August 1974	
		13. NUMBER OF PAGES 152	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED	
		15a. DECLASSIFICATION/ DOWNGRADING SCHEDULE	
15. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.			
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
18. SUPPLEMENTARY NOTES <div style="text-align: center;">Reproduced by NATIONAL TECHNICAL INFORMATION SERVICE U S Department of Commerce Springfield VA 22151</div>			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) moment invariants weighted k-nearest neighbor decision rule feature vectors adaptive training autocorrelation silhouette facial recognition			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) A system for the recognition of human faces from full profile silhouettes is described. The system is adaptively trained using a novel stack-oriented training procedure which is shown to be quite effective in identifying those feature vectors which are of most importance in the recognition process. Thus the training procedure generally produces authority files having a small number of entries. The feature vectors used are generated from a normalized autocorrelation function expressed in polar coordinates. These feature vectors are shown to be more effective in the recognition process than are the			

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

20. Abstract (Continued)

moment invariant features, at least for this problem. Experiments are described in which the system attains a recognition accuracy of 90% in a 10 class problem using 12-dimensional circular autocorrelation feature vectors. It is shown, by further experiments, that these results are no worse than a human observer's accuracy.

ia
UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

COMPUTER RECOGNITION OF FACIAL PROFILES

A Thesis

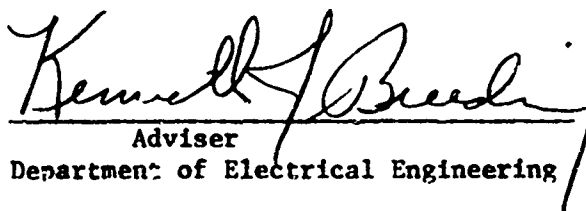
Presented in Partial Fulfillment of the Requirements
for the Degree Master of Science

by

Gerald J. Kaufman, B.S.

The Ohio State University
1974

Approved by


Adviser
Department of Electrical Engineering

if

"How can a computer be made to recognize a human face? This question remains unanswered, because pattern recognition by computer is still too crude to achieve automatic identification of objects as complex as faces."

Leon D. Harmon
November, 1973

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my adviser, Professor Kenneth J. Breeding, for his guidance, support, and understanding. The basis for this thesis, the suggestion that the state of the art in pattern recognition was sufficient to enable a machine capable of recognizing human faces to be built, was his. The automatic training algorithms described in Chapter III and upon which this work hinges were also his. Professor Breeding not only restrained me from traveling too far down cul-de-sacs, but also listened patiently to a multitude of ideas (of varying worth) and offered constructive criticism of even the least useful. He suggested several of the experiments described in this thesis. Our discussions added immensely to my knowledge and understanding of the practical as well as theoretical aspects of pattern recognition.

The curiosity and support of Mr. Thomas Southard must be mentioned. Not only was his optimism when things were not going well appreciated, but in explaining the work to him I clarified the concepts to myself.

I would like to thank my ten volunteers who submitted without complaint to my lights and camera. Their patience and cooperation made this thesis possible and their contribution entitles them to at least mention by name. They are: Ms. Carol Buccilla, Prof. Hooshang Hemami, Ms. Julie Kauper, Ms. Kathy Mielke, Ms. Gloria Peloso, Mr. Gene Rissanen,

Mr. Craig Robinson, Mr. Thomas Southard, Prof. John Swartz, and Mr. Gerald Takasaki. In addition, Ms. Kauper, Mr. Robinson, and Mr. Southard attempted the human profile recognition task described in Section 3.1.

The work of Mr. Henry Pageau in the preparation of the photographs used in this thesis is gladly acknowledged.

Ms. Julie Kauper was responsible for translating my illegible and decidedly minute scrawl into typed copy. Her efforts are most gratefully appreciated.

This work was supported in part by The Air Force Office of Scientific Research under Grant AFOSR-71-2048.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	iii
LIST OF TABLES	vii
LIST OF FIGURES	ix
Chapter	Page
I INTRODUCTION	1
1.1 Problem Formulation	1
1.2 Survey of Facial Recognition	4
II PATTERN RECOGNITION TECHNIQUES	11
2.1 Introduction	11
2.2 System Description	16
2.3 Experimental Procedure	17
2.4 Image Filtering	18
2.5 Binary Image Transformations	21
2.6 Classification Algorithms	32
III FACIAL RECOGNITION AND AUTOMATIC TRAINING . . .	37
3.1 Facial Profile Recognition	37
3.2 Automatic Training	51
3.3 Two Training Algorithms	52
3.4 An Experimental Comparison	54
3.5 A Large Sample Size Experiment	58
IV SYSTEM OPTIMIZATION	66
4.1 Introduction	66

TABLE OF CONTENTS (Continued)

Chapter	Page
4.2 Circular Autocorrelation	67
4.3 Comparison of Some Classification Algorithms	79
V SUMMARY	85
5.1 Summary of Results	85
5.2 Extensions of This Research	89
APPENDIX A: Selected Facial Profiles	92
APPENDIX B: Selected Profiles after Edge Extraction . .	97
APPENDIX C: Computer Programs	102
REFERENCES	137

LIST OF TABLES

Table	Page
3.1 Facial Profile Recognition Experiment 1 Results . . .	39
3.2 Facial Profile Recognition Experiment 2 Results . . .	39
3.3 Facial Profile Recognition Experiment 3 Results . . .	41
3.4 Facial Profile Recognition Experiment 4 Results . . .	41
3.5 Facial Profile Recognition Experiment 5 Results . . .	42
3.6 Typical Moment Invariant Feature Vectors.	46
3.7 Moment Invariants Recognition Experiment 1 Results. .	47
3.8 Moment Invariants Recognition Experiment 2 Results. .	47
3.9 Moment Invariants Recognition Experiment 3 Results. .	48
3.10 Human Facial Profile Recognition Accuracy	50
3.11 Sort Algorithm Recognition Accuracies	55
3.12 Sort Algorithm Recognition Accuracies per Class . . .	57
3.13 Activity Sort Rule on a Large Training Set	59
3.14 Training Algorithm Statistics	61
4.1 Circular Autocorrelation Feature Vectors under Image Rotation and Size Change for 9" x 12" Rectangle . . .	70
4.2 Recognition Accuracy for Selected Circular Auto- correlation Function Parameters	72
4.3 Recognition Accuracy per Class for Selected Circular Autocorrelation Function Parameters	72
4.4 Typical Feature Vectors for Selected Circular Auto- correlation Function Parameters	73
4.5 Feature Vectors for Circular Autocorrelation Function Parameters $a_m = 1/2$, $M=1$, $N=12$	75

LIST OF TABLES (Continued)

Table	Page
4.6 Weight Function Comparison for the Distance Weighted k-Nearest Neighbor Rule	82
4.7 Recognition Accuracy for Two Classifier Types	82
4.8 Recognition Accuracy per Class for Two Classifier Types	82
4.9 Feature Vector Component Means and Standard Deviations	83

LIST OF FIGURES

Figure	Page
2.1 The Basic Two-Dimensional Pattern Recognition Machine .	12
2.2 The Nearest Neighbor Rule Decision Surface.	34
3.1 Recognition Accuracy vs. Authority File Depth	43
3.2 Number of Vectors in Authority Files vs. Input Samples	63

CHAPTER I

INTRODUCTION

1.1 Problem Formulation

The use of computers for the recognition of two-dimensional images has been the subject of theoretical and experimental research for over a decade [1,2,3,4]. Originally spurred by the problem of character recognition for computer input, researchers have recently begun [5,6,7] to branch out and consider the recognition of other, more complex, two-dimensional images. This thesis describes an attempt to apply a portion of the large body of pattern recognition theory to the problem of machine recognition of human faces. Such a machine could have obvious applications in many personal identification or verification roles. Applications in law enforcement, credit verification, security systems, and surveillance come to mind. This thesis is mostly experimental in nature, its purpose being to select the best pattern recognition techniques for the problem and assemble them into a working system.

The problem may then be stated as follows: To demonstrate that a system capable of recognizing humans from their facial images (such as would be obtained by a television camera) in real time with an acceptably low error rate is possible using presently available hardware and pattern recognition techniques. The ultimate goal of this work might then be

to have a television camera viewing the entrance to a restricted area with the video output fed to a computer. If the computer is able to identify faces, then it can perform a table look-up to find if a person requesting entrance is authorized and take appropriate action (e.g., opening the door, or calling the security guard). Since this thesis is concerned only with demonstrating that such a machine is possible, existing hardware was used and the problem was simplified as much as possible. The details of the problem follow.

It was decided to define a ten class problem, that is, ten people were chosen to comprise the input set. This number, although too small for most practical applications, does provide a simple starting point. The small number of classes allows the data generated to be analyzed without time consuming calculations and also allows the input data to be gathered in a reasonable period of time. By character recognition standards, a ten class problem may not be large enough to provide a fair test of the classification system. The work of Goldstein, Harmon, and Lesk [8] indicates, however, that for facial recognition, a ten class problem may be sufficient to provide some indication of how the system would respond to a larger problem. Again, it is not the intent of this thesis to provide a practical pattern recognition system for human faces, only to demonstrate its feasibility. The ten subjects may be broken down into the following categories: four were female, six were male, two wore glasses with dark plastic frames, two wore wire rimmed glasses. The two-dimensional image was obtained from a

television camera output interfaced directly to the computer [9]. The video signal was quantized to only two levels, black or white, with the quantization threshold level set by the operator. Because of this severe quantization it was felt (after some preliminary experimentation) that the profile view offered the most information and maximum repeatability from image to image. Profile views were thus used exclusively in this work. A total of 120 images, 12 per subject, were taken and stored on magnetic tape for processing. These 120 images comprised the input set. The machine was designed to classify an input image as belonging to one of the ten classes and output its classification. The possibility of an input image not belonging to one of the ten classes was ignored in the interest of simplicity.

The rest of this chapter contains a survey of the work done in the area of human face recognition, both by computer and human recognizers. Chapter II provides a summary of the more prominent two-dimensional pattern recognition techniques, with emphasis on the techniques investigated in this work. Chapter III discusses the results of machine and human recognition on the 120 facial profiles. Chapter III also describes two algorithms used to train the pattern recognition system and experiments to verify their expected operation. Chapter IV gives a description of the series of experiments performed to optimize the pattern recognition techniques for the facial recognition problem. Chapter V concludes the thesis with a summary of the results and a discussion of areas for further research.

1.2 Survey of Facial Recognition

Virtually all of the previous work on the problem of facial recognition has dealt with syntactical information as may be found in a set of roughly quantized descriptive features such as ear length, lip thickness, chin profile, etc. [10,11]. No attempt to use statistical recognition techniques [12] for the identification of faces has been found.

The first work with syntactic recognition is that of Bertillon [11] in the classification of facial features for criminological application. Although later superseded by his work in fingerprint classification, his facial descriptions were meticulously done and included sets of descriptive names still used by law enforcement agencies. A more modern discussion may be found in Allen [13].

The closest to an automatic recognition system is a man-machine interactive approach described by Goldstein, Harmon and Lesk [10], and Harmon [14]. This system used a 21-dimensional feature vector. The vector components were descriptive features quantized on a scale of one to five. Some examples of the features are: mouth width (short to long), cheeks (sunken to full), and hair length (short to long). The input set consisted of 255 faces and the values of the feature vector components were determined by the average of assignments made by a panel of ten human observers from three photographs (front, 3/4, and side views) of each face. The feature vectors were entered into the computer and a sorting algorithm used to order the vectors from best to worst

match for some input description. The input description was obtained by first having the operator enter the most "conspicuous" features of the subject to be identified and then allowing the computer to request features that would separate the vectors at the top of the rank-ordered list. The procedure was stopped after ten of the 21 feature vector components had been entered. Using this technique a recognition accuracy of 70% was achieved.

Kaya and Kobayashi [15] suggested that a set of geometric parameters could be used to describe a human face. They defined nine parameters that were Euclidian distances between specified points on the front view of a face. Some typical parameters are height of lips, distance between upper lip and nose, distance between lower lip and chin, and distance between corners of the eyes. All parameters were normalized by the nose length to provide size invariance. These parameters were measured from a set of photographs of 62 people. A serial classification or tree search algorithm was proposed in which each parameter is used in turn to reduce the population until only one face remained. A theoretical analysis using assumed parameter probability distributions and negligible noise showed that the algorithm could achieve a recognition accuracy of 90% within a population of 15,000 faces.

The other investigations of the recognition of human faces are concerned with recognition by humans. Goldstein, Harmon, and Lesk [8] describe a series of experiments leading to 22 subjective features of human faces that are useful for recognition. The sample set consisted

of Harmon's 255 faces, and ten "jurors" were used to assign numerical values to the features. The 22 features were selected from a set of 34, the selection criteria being large variance over the sample set and small variance over the jurors. The 22 features were tested for correlation and found to be largely independent by several tests. A model for classification by humans was proposed in which the features were ordered from most "extreme" to least "extreme". Each feature was selected in turn and faces with a feature value close enough (by some constant threshold) to the specified feature value were kept, while the rest were rejected. The number of features used to narrow the sample set to one using this technique was found to depend logarithmically on the size of the sample set and an equation describing this dependence was derived. Approximately six features for the 255 samples were used and it was predicted using the previously derived equation that about 14 features would be necessary for a sample size of 4×10^6 . The computer model was verified with an experiment using human recognizers who obtained a recognition accuracy of 53%.

Goldstein and Mackenberg [16] experimented with facial recognition by humans given only a portion of the face. The recognizer was required to identify a known person from a photograph which had been masked so that only a portion of the face was visible. This study indicated that for recognition the upper parts of the face are more important than the lower parts. Recognition accuracy was generally better for pictures that contained several of Harmon's 22 features than for pictures that

contained few.

Harmon [17] reported an investigation into the minimum amount of information necessary for facial recognition by humans. Fourteen front view photographs were digitized with a flying-spot scanner and stored on magnetic tape. The high quality image (1024 x 1024 bits) was fragmented into $n \times n$ squares and each square was assigned a brightness equal to the average of the brightness values of all the original samples within the square. Brightness was quantized to 8 or 16 levels. The final picture thus obtained contained high frequency noise corresponding to the block edges and although the energy content of the high frequencies was small, recognition was improved by low pass filtering. It was hypothesized that this was due to the eye's sensitivity to straight lines and regular geometric shapes. With the facial image quantized to a 16 x 16 grid with 8 grey levels, an average recognition accuracy of 48% was achieved by 28 human recognizers. The results also indicated that the grid placement on the photographs may be critical for optimum recognition. Harmon and Julesz [18] used the same technique to investigate the effect of noise on the recognition of faces. They found that random noise at frequencies close to the spatial quantization frequency "masked" the information and raised the amount of information necessary for recognition, while random noise of frequencies greater than two octaves removed from the quantization frequency had little effect upon recognition.

Hochberg and Galper [19] tested the perception of, and memory for, faces by humans. They found that recognition accuracy was

significantly greater for upright than for inverted photographs of human faces. Faces that were learned from upright photographs seemed to be tied to orientation, while faces learned from inverted photographs did not seem to be tied strongly to orientation. Bradshaw and Wallace [20] used an Identi-Kit to obtain information on how humans recognize faces. Their results indicated that humans classify faces with a "serial self-terminating" procedure, that is, each facial feature is considered in turn until an identification is made, at which point the procedure stops. They found no evidence for parallel processing, or Gestalt perception [21].

If pattern recognition is one side of a coin, then pattern generation is the other (and usually more tractable) side. Two efforts at the computer generation of faces, although not terribly germane to the present work, are interesting enough to be mentioned. Gillenson [22] designed a system for use by "non-artists" to reconstruct a line drawing of the front view of a face on a cathode ray tube display. The system was interactive with the user and consisted of a library of stored features with routines to distort the features to obtain a better likeness. Parke [23] developed a system to draw high quality half-tone renderings of a human face in three-dimensional perspective. The skin surface was approximated by a net of polygons and a shading algorithm was used to give a continuous curved appearance. The face was represented by a matrix describing the polygon vertices, and animation was achieved by interpolating between the vertices' positions for two end expressions.

In addition to the above, some qualitative work may be briefly mentioned. The Identi-Kit is the best known semi-automatic system for generating line drawings of the front view of a human face and is used mainly by police departments. The Identi-Kit uses clear plastic overlays each having a single facial feature to form a composite picture. There are several different overlays for each feature, so that by choosing the appropriate features a reasonable facsimile of a specific human face may be produced. There are several other similar systems in use, and a summary of their differences and operation may be found in Gillenson [22].

Wall [24] indicates that criminological eyewitness identification is virtually always subject to error unless the witness knew the subject in advance of the act. The factors of fright, similarity of faces, and poor and fast viewing conditions tend to make accurate recognition difficult. Although artists have been drawing human faces for all of history, there is little in artistic literature discussing the recognition of faces. Willis [25] mentions the different variations of facial features and how they may be realistically represented in drawings.

The above survey points out the basic differences between the previous work in facial recognition and the work described in this thesis. This thesis describes the development of a completely automatic and real time facial recognition system. This system uses statistical pattern recognition techniques to classify facial profiles obtained from a television camera input. Previous systems have been at best man-machine interactive with the facial features described to the machine by the human observer from a photograph. Previous work has also depended upon

syntactic pattern recognition with no investigation into the applicability of statistical pattern recognition to the problem. Finally, the facial recognition system described in this thesis provides the highest recognition accuracy of any system known to the author.

CHAPTER II

PATTERN RECOGNITION TECHNIQUES

In this chapter the pattern recognition aspects of this research are discussed. A short review of the two-dimensional pattern recognition problem is provided, followed by a description of the hardware and some of the software involved in this particular system.

2.1 Introduction

Although the term pattern recognition encompasses far more than picture recognition, in the interest of conciseness, only the techniques associated with two-dimensional image recognition and germane to this work will be mentioned. Most designers of digital two-dimensional pattern recognition machines seem to use a rather standard structure. This basic form is shown in Figure 2.1. Most workers in the image recognition area ignore the problem of target acquisition. The scene presented to the machine contains only the pattern to be recognized, a simplification which may not be realistic outside the laboratory environment.

The input device to the image recognition system may be non-existent (i.e., the pattern is digitized by hand and entered through a standard peripheral), an array of photocells, a flying spot scanner, or television camera. The input device is usually responsible for the

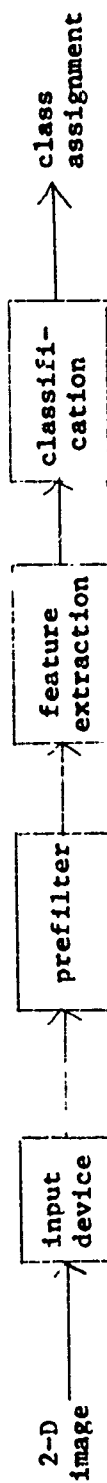


Figure 2.1. The Basic Two-Dimensional Pattern Recognition Machine

image digitization. Prefiltering of the image may or may not be used. Prefiltering is used to remove any extraneous noise from the image, and, in some cases, smooth the image boundary.

The process of feature extraction, or image transformation, is used to reduce the amount of information to be handled by the classifier by extracting "features" from the image that are in some sense representative of the image. It is desirable that these features be invariant with respect to image position (translation), size change, and rotation, so that an image which is modified by any combination of the above three transformations will be classified the same as the original image. Several techniques for feature extraction from two-dimensional images have been developed. The earliest used are the related techniques of cross-correlation, template matching, and matched filtering [1,26]. These methods compare a stored pattern against the input image and produce a single metric which is related to the 'goodness of match'. These techniques have several disadvantages. Since comparison patterns (templates) must be stored, the machine memory size must be very large even for simple recognition problems consisting of only a few classes with small image sizes (large quantization intervals). In general these techniques are not size or rotation invariant.

Another technique is that of geometric features extracted by local neighborhood operations in array processors. First proposed by Unger [27,28], the work has been extended by several other researchers [29,30]. The major problem with this technique seems to be

its lack of generality, although it has been shown to work for simple geometric shapes [31], it has not been successfully applied to the recognition of complex objects. The features are usually not invariant with respect to size or rotation.

A class of feature extraction techniques that seems to be gaining support at present are the various frequency domain algorithms. Among these are the impulse response filters [32], the discrete Fourier transform, and the Walsh/Hadamard/Haar transforms. The impulse response filters are often used in the analysis of aerial photographs and terrain classification [33] to detect small areas of interest such as orchards, oil tank farms, and railroad yards. The filters are simply a distribution of integer weights on a grid such that when the appropriate feature (e.g., a straight line) is centered under the grid the sum of all weights times their associated optical densities is above a threshold. The discrete Fourier transform has given good results on a number of pattern recognition problems [6]. The use of digital transform domains such as the Walsh-Hadamard and Haar has been proposed [34]. These techniques generally suffer from the same lack of invariance as the other methods discussed above, i.e., size and rotation, although Richard's technique of using Fourier descriptors of the boundary curve of an image [7] is both size and rotation invariant.

The last class of techniques is that of arbitrary transformations. These are transforms designed to be invariant with respect to translation, size change, and rotation. Circular auto-correlation and moment

invariants [35] are in this class. Moment invariants have been used by Dudani [5] with excellent results. Circular auto-correlation was developed during this research and will be described below.

The final process in a pattern recognition system is classification. Usually the unknown feature vector is compared against a list of vectors for each class (authority files) and some metric for each class, corresponding to the probability that the unknown vector belongs to that class, is computed. The unknown vector is then assigned to the class to which it has the highest probability of belonging.

A probabilistic classifier of the Bayes type will theoretically give the highest possible recognition accuracy on a given set of feature vectors [36]. The drawback to the Bayes classifier is the need to know the a priori probability density functions for the occurrence of a vector and the occurrence of a vector given that it belongs to a specified class. These functions are seldom, if ever, known. The experimental determination of such functions requires large sample sizes usually not available to the researcher. Without accurate probability density functions the Bayes classifier may not perform as well as a non-probabilistic classifier [5].

A simple piecewise-linear classifier is the nearest neighbor classifier. In this method the unknown feature vector is assigned to the class to which it has the smallest Euclidian distance. It has been shown that this technique will produce a recognition accuracy no worse than one-half that of an optimal (Bayes) classifier [37].

A more complex and non-linear classifier is the distance-weighted k-nearest neighbor rule [5,37]. This technique assigns weights to the k nearest neighbors from the authority file of the unknown vector. The weights assigned to the k nearest neighbors are summed with respect to class and the unknown vector assigned to the class with the highest weight.

The vectors used in a classifier may be either normalized or unnormalized. One method of normalization is to divide all vectors by their length, so that only vector angles determine classification. Another method is to subtract from each component of the vector that component's mean and then divide by its standard deviation [38]. The type of vector normalization used, if any, depends upon both the feature extraction and classification algorithms.

2.2 System Description

The hardware used in this experiment will now be described. The computer used to implement the recognition algorithms is the Ohio State University Electrical Engineering Department's PDP-9. This particular machine configuration contains, in addition to the standard peripherals, a Tektronix 611 bistable storage cathode ray tube display. Connected to the PDP-9 is a closed circuit black and white television camera. The interface [9] which connects the camera to the computer contains a circuit to threshold the video input signal from the camera to a binary output. The threshold level is adjustable and the binary video signal is fed to a television monitor so that the operator may

adjust the threshold level to compensate for changing light levels and obtain the desired image. The images obtained from this system are 360 x 240 bit binary arrays and are stored in the machine as 20 x 240 18-bit word arrays. The interface actually reads only about 180 bits per scan line, which gives a distorted image when displayed on an array with equal horizontal and vertical bit spacing. An aspect ratio correction routine is therefore employed [9] to approximately double the number of bits horizontally, which results in an undistorted image.

2.3 Experimental Procedure

The procedure used to obtain a facial profile consisted of seating the subject in front of a black backdrop facing perpendicular to the optical axis of the television camera. The camera position was adjusted so that the subject's profile filled the monitor screen. The lighting, video threshold, and camera aperture were selected to obtain a reasonable replica on the monitor. In this system, subjects' flesh appeared white on the monitor and the hair and background were black. Appendix A contains samples of negatives of the facial profiles obtained. Subjects 8 and 10 wore horn rimmed and black frame glasses, respectively, and as can be seen from Appendix A, these glasses were thresholded as black. Subjects 5 and 6 wore wire rimmed glasses, which are not visible on the thresholded image.

Once a satisfactory image was obtained on the monitor, the image was stored on a disk file and later transferred to DECtape. The idea here was to build a large file of facial images so that it would not be

necessary to have the subjects present every time an identification system was to be tested. With this file of images, it was then possible to divide it into two distinct sets, one for training the recognition system and one for testing the recognition accuracy with unknown profiles.

It was the author's intention at the beginning of this work to obtain one image per day for each of the ten subjects. It was felt that this procedure would give a set of images with 'real' day to day variations and thus be a reasonable data set upon which to base a pattern recognition system. This dream was shattered by the practical problem of scheduling computer time and matching personal schedules. In the end, most of the images were obtained in two sittings of six images each per subject, for a total of 120 images. This total data set may be divided into two sets of 60 images, 6 images per subject. The first 60 images were obtained with changes in lighting, video threshold and camera aperture settings, and subject position in an attempt to introduce variations in samples within a given class. The second set of 60 images was obtained with fixed lighting, video threshold, camera aperture, and subject position in an attempt to reduce the sample variation to a minimum.

2.4 Image Filtering

The pattern recognition system described in this thesis uses two image filtering routines. The first is a 'prefilter' routine whose purpose is to remove high frequency noise on the image boundary. This noise is caused by several factors. First is the quantization noise

inherent in any digitization process. Second is the noise generated by the aspect ratio correction mentioned previously. Because the number of horizontal bits is almost doubled, steps in the image boundary two cells wide tend to occur. Third is the noise generated by inherent instabilities in the video level and threshold circuits. The video signal from the television camera is not very stable and consequently it is difficult to obtain a smooth boundary on the thresholded image for a light to dark transition in the viewed scene. This effect is most apparent in the hairline area of the facial profiles, as can be seen from the pictures contained in Appendix A. The second filter is used to extract the front edge of the facial profile. This filter rejects the larger image variations caused by the highly variable hairline area described below.

The prefilter is implemented with binary array processor simulation routines [39]. Briefly, the array processor structure of this simulation is a synchronous two-dimensional array of storage cells. Each cell contains one bit of a binary image. The next state of each cell is a binary function of its present state and the states of the eight "neighbor" cells adjoining it (the simulation routines use a rectangular tessellation [40]). Threshold logic is used to implement the binary function. The algorithm for the prefilter is to first set to zero any cell whose eight neighbors are not all one, thus removing "noise" cells. Then any cell that is zero and has at least one neighbor that

is one on a Von Neumann neighborhood [41] is changed to one. Finally, any zero cell that has at least one neighbor that is one on a Moore neighborhood [41] is changed to one. The last two operations have the effect of smoothing the image boundary as well as filling one and two-cell-wide gaps in the image. The boundary smoothing is particularly useful in this system because the aspect ratio correction [9] required by the television camera interface tends to generate two-cell-wide steps in the input image.

The edge extraction filter is used to remove the back of the facial profile. It was determined early in this work that the television camera and threshold circuit used to obtain the binary images gave drastically different images in the hairline and chin-neck areas for slight lighting changes. It was also felt that the hairline and collar areas would be highly variable on a day to day basis. For these two reasons it was decided to mask all but the front of the facial profile and use only this edge for classification. The edge extraction is accomplished by duplicating the input image, shifting the duplicated image backwards, and then setting to zero all image cells covered by one cells of the duplicated array. The second array is then moved up and down with the image cells overlayed by the one cells of the second array being set to zero after each move. The distance of all translations is given by

$$d = K \sqrt{A} \quad (2-1)$$

where: d is the translation distance of the duplicated array

A is the total image area, i.e., the number of one cells

K is a constant empirically determined.

This relation tends to make the routine size independent (i.e., the same height-to-width ratio of the filtered image is maintained). This is not a very accurate method of obtaining size independence, since two images of a subject may have significantly different areas, due to the input system flaw mentioned above (see Appendix A). The method did, however, seem to be adequate for this system. This filter routine is certainly not the only, or even best, method of obtaining this function, since it leaves the forehead area dependent upon the hairline (see Appendix B). It does have the advantages of being simple, fast, translation and size invariant, and useable for any rotation angle. The binary array processor simulation routines used in this pattern recognition scheme could have been used to obtain an image edge simply by setting to zero all cells with the proper neighborhood. For example, to obtain the right edge of an image any cell whose righthand neighbor is one should be set to zero. This method of edge extraction was not used for two reasons. It was felt that the binary image transformations would be less susceptible to noise if they were presented with a solid image rather than a single-cell-wide edge. Also, this edge extraction technique can only extract edges at 45° increments, which is undesirable if the system is to be expanded to work with arbitrary image rotations.

2.5 Binary Image Transformations

This research uses two of the image transformation techniques previously discussed, correlation and moments. Correlation was chosen

mainly for its ease of computation and its ability to be made translation and size invariant, and have predictable and easily computed variation under rotation. Although correlation does not presently seem to be a popular technique with researchers in the pattern recognition area, it has been used by Horwitz and Shelton [42] on a simple character recognition experiment with good results. Moment invariants were chosen because of the excellent results Dudani [5] obtained on aircraft recognition with this technique. It was felt that moments would provide a reasonable benchmark to compare correlation against, as well as provide information on how well this transformation performs on a different set of binary images with an increased number of classes.

Consider then a finite, continuous, binary, two-dimensional image, I , on some plane, P . For any point p in P , I is assigned the value ϕ or 1 . Introducing a cartesian coordinate system in P with coordinates (x,y) allows I to be written as a function of x and y :

$$f(x,y) = 0,1 \quad \text{for any real } x,y \quad \begin{matrix} -\infty < x < \infty \\ -\infty < y < \infty \end{matrix} \quad (2-2)$$

The plane is infinite in extent, but we will require I to be finite, that is, for the image area defined as:

$$A = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) \, dx \, dy \quad (2-3)$$

(which is simply the number of one cells for a binary image), we require A to be finite; $A < \infty$.

The image may be digitized with a two-dimensional sampling function:

$$f(m\gamma, n\gamma) = f(x, y) \delta(x - m\gamma) \delta(y - n\gamma) \quad (2-4)$$

for m and n integers, $-\infty \leq m \leq \infty$
 $-\infty \leq n \leq \infty$.

$\delta(x)$ is the impulse function, defined as

$$\begin{aligned} \delta(x) &= 1, \quad x = 0 & -\infty \leq x \leq \infty \\ \delta(x) &= 0, \quad \text{otherwise} \end{aligned} \quad (2-5)$$

and γ is the sampling interval (a real constant). The effect of digitization is simply to add a quantization noise, η , to I and any function of I :

$$\eta(x\gamma, y\gamma) = f([x\gamma], [y\gamma]) - f(x\gamma, y\gamma) \quad (2-6)$$

where $[x]$ is the greatest integer function.

The mean error is then given by:

$$\bar{\epsilon} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \eta(x, y) dx, dy. \quad (2-7)$$

The quantization noise is usually reduced to a tolerable value simply by setting γ much smaller than the size of any portion of I that is of interest. Rosenthal has shown [43] that any function of a digitized image may be expressed as a continuous function of the continuous image to within some quantization error. In the following discussion this approach will be taken since it results in a somewhat simpler and, in the author's opinion, more elegant formulation. In the pattern recognition work described in this thesis, the input system noise and the

image variations within a given class are much larger than the quantization error, so no analysis of the quantization error was performed.

Consider the two-dimensional auto correlation of I:

$$g(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) f(x+u, y+v) dx dy \quad (2-8)$$

for u,v real. $-\infty \leq u \leq \infty$, $-\infty \leq v \leq \infty$.

Using the identities

$$u = \sqrt{A} \alpha \cos \theta \quad \text{for } \theta, \alpha \text{ real, } 0 \leq \theta \leq \pi \quad (2-9)$$

$$v = \sqrt{A} \alpha \sin \theta \quad 0 \leq \alpha \leq \infty \quad (2-10)$$

the autocorrelation function of I may be expressed in a size normalized polar coordinate form:

$$g(\alpha, \theta) = \frac{g(u,v)}{A} \quad (2-11)$$

The factors A^{-1} and $A^{1/2}$ are used to obtain size invariance of $g(\alpha, \theta)$. The autocorrelation function is even in u and v , which translates into periodicity in the polar coordinates with period π . This may be demonstrated by setting:

$$\alpha' = \alpha \quad (2-12)$$

$$\theta' = \theta + \pi \quad (2-13)$$

Then,

$$A' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,v) dx dy = A \quad (2-14)$$

$$u' = \sqrt{A'} \alpha' \cos \theta' = \sqrt{A} \alpha \cos(\theta + \pi) = -\sqrt{A} \alpha \cos \theta = -u \quad (2-15)$$

$$v' = \sqrt{A'} \alpha' \sin \theta' = \sqrt{A} \alpha \sin(\theta + \pi) = -\sqrt{A} \alpha \sin \theta = -v \quad (2-16)$$

so

$$\begin{aligned} g(u', v') &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) f(x+u', y+v') dx dy \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) f(x-u, y-v) dx dy \quad . \end{aligned} \quad (2-17)$$

Now using the substitution,

$$x' = x - u \quad (2-18)$$

$$y' = y - v \quad (2-19)$$

$$dx' = dx, \quad dy' = dy$$

we have

$$g(u', v') = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x'+u, y'+v) f(x', y') dx' dy' = g(u, v) \quad (2-20)$$

therefore

$$g(\alpha', \theta') = \frac{g(u', v')}{A} = \frac{g(u, v)}{A} = g(\alpha, \theta) \quad (2-21)$$

and thus $g(\alpha, \theta)$ is periodic in θ with period π .

The autocorrelation function is also invariant under translation, since with

$$x' = x + a \quad (2-22)$$

$$y' = y + b \quad \text{for } a, b \text{ arbitrary real constants } -\infty < a < \infty, -\infty < b < \infty. \quad (2-23)$$

$$A' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x+a, y+b) dx dy \quad (2-24)$$

$$g'(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x+a, y+b) f(x+u+a, y+v+b) dx dy \quad (2-25)$$

Changing the variables of integration to x' and y'

$$dx' = dx \quad dy' = dy$$

$$A' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') dx' dy' = A \quad (2-26)$$

$$g'(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') f(x'+u, y'+v) dx' dy' = g(u, v) \quad (2-27)$$

and then also

$$g'(\alpha, \theta) = g(\alpha, \theta) \quad (2-28)$$

The size normalized polar form of the autocorrelation function is invariant under image size change. To show this, let,

$$x' = ax \quad \text{for } a \text{ an arbitrary real constant} \quad (2-29)$$

$$y' = ay \quad (\text{the magnification}) \quad 0 < a \leq \infty \quad (2-30)$$

$$A' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(ax, ay) dx dy \quad (2-31)$$

$$g'(u', v') = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(ax, ay) f[a(x+u)a(y+v)] dx dy \quad (2-32)$$

Changing the variables of integration to x' and y'

$$dx' = adx \quad dy' = ady$$

$$A' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') a^2 dx' dy' = a^2 A \quad (2-33)$$

$$\begin{aligned} g'(u', v') &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') f(x'+au, y'+av) a^2 dx' dy' \\ &= a^2 g(au, av) \end{aligned} \quad (2-34)$$

and

$$u' = \sqrt{A'} \alpha \cos \theta = \sqrt{a^2 A} \alpha \cos \theta = a\sqrt{A} \alpha \cos \theta = au \quad (2-35)$$

$$v' = \sqrt{A'} \alpha \sin \theta = av \quad (2-36)$$

so

$$g'(\alpha, \theta) = \frac{g'(u', v')}{A'} = \frac{a^2 g(au, av)}{a^2 A} = g(\alpha, \theta) \quad (2-37)$$

The size normalized polar form of the autocorrelation function is not invariant under image rotation, but it does change by only a phase factor equal to the image rotation angle. This can be seen using the identities:

$$x' = x \cos \phi - y \sin \phi \quad \text{for rotation angle } \phi \quad (2-38)$$

$$y' = x \sin \phi + y \cos \phi \quad 0 \leq \phi \leq 2\pi \quad (2-39)$$

$$A' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x \cos \phi - y \sin \phi, x \sin \phi + y \cos \phi) dx dy \quad (2-40)$$

$$g'(u', v') = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x \cos \phi - y \sin \phi, x \sin \phi + y \cos \phi) \\ \cdot f[(x+u) \cos \phi - (y+v) \sin \phi, (x+u) \sin \phi + (y+v) \cos \phi] dx dy \quad (2-41)$$

Changing the variables of integration to x' and y'

$$\frac{\partial x'}{\partial x} = \cos \phi, \quad \frac{\partial x'}{\partial y} = -\sin \phi$$

$$\frac{\partial y'}{\partial x} = \sin \phi, \quad \frac{\partial y'}{\partial y} = \cos \phi$$

$$|J| = \begin{vmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{vmatrix} = \cos^2 \phi + \sin^2 \phi = 1$$

$$A' = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') dx' dy' = A \quad (2-42)$$

$$g'(u', v') = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') f[x' + (u \cos \phi - v \sin \phi), y' + (u \sin \phi + v \cos \phi)] dx' dy' \\ = g(u \cos \phi - v \sin \phi, u \sin \phi + v \cos \phi) \quad (2-43)$$

and if we let

$$\alpha' = \alpha \quad \theta' = \theta + \phi \quad (2-44)$$

$$u' = \sqrt{A} \alpha \cos(\theta + \phi) = \sqrt{A} \alpha (\cos \theta \cos \phi - \sin \theta \sin \phi) \\ = u \cos \phi - v \sin \phi \quad (2-45)$$

$$v' = \sqrt{A} \alpha \sin(\theta + \phi) = \sqrt{A} \alpha (\sin \theta \cos \phi + \cos \theta \sin \phi) \\ = u \sin \phi + v \cos \phi \quad (2-46)$$

$$\text{so } g(\alpha', \theta') = \frac{g'(u', v')}{A} = g(\alpha, \theta + \phi). \quad (2-47)$$

The function $g(\alpha, \theta)$ may be used to transform a binary image into an MN -dimensional vector by setting:

$$\alpha = a_m \quad \text{with } a_m \text{ a real constant} \quad (2-48)$$

$$m, n, M, N \text{ integers}$$

$$\theta = \frac{\pi(n-1)}{N} \quad \begin{matrix} 1 \leq m \leq M \\ 1 \leq n \leq N \end{matrix} \quad (2-49)$$

For lack of a more inspired name this transformation is referred to as the circular autocorrelation function. The MN terms of this function are, as shown above, independent of translation and size changes of the input image, I (within, of course, quantization error for a discrete computation). Under an image rotation of π/N radians, the terms are related by

$$g'(m, 1) = g(m, N) \quad (2-50)$$

$$g'(m, n) = g(m, n-1), \quad n > 1 \quad (2-51)$$

and similarly for other increments of π/N . It can be seen that for rotation angles other than $\pi n/N$ there will be another quantization error introduced, and thus N must be chosen large enough to make this error negligible for the particular application. This relation between image rotation and the circular autocorrelation term sequence allows the image angle to be determined to within π/N radians (\pm a π radian ambiguity because of $g(\alpha, \theta)$'s periodicity in θ of π). The penalty for this feature is an increased classification time over a rotation invariant transformation because of the N searches of the authority files

required. The ability to determine image rotation angle, however, may be worth the time trade-off in many applications. Practically, the circular autocorrelation function is computed simply by duplicating the input image, shifting the duplicated image a distance $\sqrt{A} \alpha$ at angle θ and counting the number of intersecting one cells.

The moment transformation is based upon a set of two-dimensional moment functions derived by Hu [35], which are invariant with respect to image translation, size, and rotation. The two-dimensional moments are defined by:

$$m_{pq} = \frac{1}{M} \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(\gamma_m, \gamma_n) m^p n^q \quad (2-52)$$

$$M = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(\gamma_m, \gamma_n) \quad p, q = 0, 1, 2, \dots \quad (2-53)$$

where: m_{pq} is the $(p+q)^{th}$ order moment
 $f(\gamma_m, \gamma_n)$ is the digitized image.

The centroid of the image is then given by:

$$\bar{m} = m_{10} = \frac{1}{M} \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(\gamma_m, \gamma_n) m \quad (2-54)$$

$$\bar{n} = m_{01} = \frac{1}{M} \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(\gamma_m, \gamma_n) n \quad (2-55)$$

where (\bar{m}, \bar{n}) are the coordinates of the centroid of $f(\gamma_m, \gamma_n)$.

The central moments are defined by:

$$\mu_{pq} = \frac{1}{M} \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(\gamma_m, \gamma_n) (m-\bar{m})^p (n-\bar{n})^q \quad (2-56)$$

The central moments may also be expressed as sums of the ordinary moments [5]; the expressions for the first three orders are:

$$\mu_{00} = m_{00} = 1 \quad (2-57)$$

$$\mu_{01} = \mu_{10} = 0 \quad (2-58)$$

$$\mu_{20} = m_{20} - (m_{10})^2 \quad (2-59)$$

$$\mu_{02} = m_{02} - (m_{01})^2 \quad (2-60)$$

$$\mu_{30} = m_{30} - 3m_{20}m_{10} + 2(m_{10})^3 \quad (2-61)$$

$$\mu_{21} = m_{21} - m_{20}m_{01} - 2m_{11}m_{10} + 2(m_{10})^2 m_{01} \quad (2-62)$$

$$\mu_{12} = m_{12} - m_{02}m_{10} - 2m_{11}m_{01} + 2(m_{01})^2 m_{10} \quad (2-63)$$

$$\mu_{03} = m_{03} - 3m_{02}m_{01} + 2(m_{01})^3 \quad (2-64)$$

The moment invariants used in this research are:

$$M'_2 = \frac{1}{r^4} [(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2] \quad (2-65)$$

$$M'_3 = \frac{1}{r^6} [(\mu_{30} - 3\mu_{12})^2 + (3\mu_{21} - \mu_{03})^2] \quad (2-66)$$

$$M'_4 = \frac{1}{r^6} [(\mu_{30} + \mu_{12})^2 + (\mu_{21} + \mu_{03})^2] \quad (2-67)$$

$$M_5' = \frac{1}{r^{12}} \{ (\mu_{30} - 3\mu_{12})(\mu_{30} + \mu_{12})[(\mu_{30} + \mu_{12})^2 + 3(\mu_{21} + \mu_{03})^2] \\ + (3\mu_{21} - \mu_{03})(\mu_{21} + \mu_{03})[3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] \} \quad (2-68)$$

$$M_6' = \frac{1}{r^8} \{ (\mu_{20} - \mu_{02})[(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] \\ + 4\mu_{11}(\mu_{30} + \mu_{12})(\mu_{21} + \mu_{03}) \} \quad (2-69)$$

$$M_7' = \frac{1}{r^{12}} \{ (3\mu_{21} - \mu_{03})(\mu_{30} + \mu_{12})[(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] \\ - (\mu_{30} - 3\mu_{12})(\mu_{21} + \mu_{03})[3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] \} \quad (2-70)$$

$$r = (\mu_{20} + \mu_{02})^{1/2} \quad (2-71)$$

It can be shown [5,35] that functions $M_2' - M_7'$ are invariant under image translation, size change, and rotation. By computing $M_2' - M_7'$ on both the image silhouette and boundary a twelve-dimensional feature vector may be obtained [5].

2.6 Classification Algorithms

Two classification algorithms were investigated in this work, the nearest neighbor rule and the distance-weighted k-nearest neighbor rule. The nearest neighbor classifier computes the Euclidian distance between the unknown vector and every vector in the authority files. The unknown vector is assigned to the class which contains the vector closest to the unknown. This may be written as:

$$e_{pq} = \left[\sum_{k=1}^n (r_k - s_k^{pq})^2 \right]^{1/2} \quad (2-72)$$

where: r_k is the k -th element of the unknown feature vector, R , of dimension n .

s_k^{pq} is the k -th element of the p -th vector S in the authority file for class q .

e_{pq} is the 'error' (Euclidian distance) between points R and S .

Then for Q authority files (and hence Q classes) each containing p vectors (depth P), there is some smallest e_{pq}

$$e_{mn} \leq e_{pq} \quad \text{for all } p, q \quad \begin{matrix} 1 \leq p \leq P \\ 1 \leq q \leq Q \end{matrix} \quad (2-73)$$

and R is assigned to class n .

The nearest neighbor rule is a simple piecewise linear classifier; that is, the class boundaries, or decision surfaces are defined by segments of hyperplanes. For example, in a two class problem with two authority vectors per class, in two dimensions, the decision surface defined by the nearest neighbor rule is shown in Figure 2.2. An unknown vector in the space to the left of the decision surface would be assigned to class 1 by the nearest neighbor rule, while an unknown vector in the space to the right of the decision surface would be assigned to class 2. Piecewise linear techniques such as the nearest neighbor rule can be combined with logical rules to construct quite complex boundaries [44], but only the simple nearest neighbor rule was used in this research.

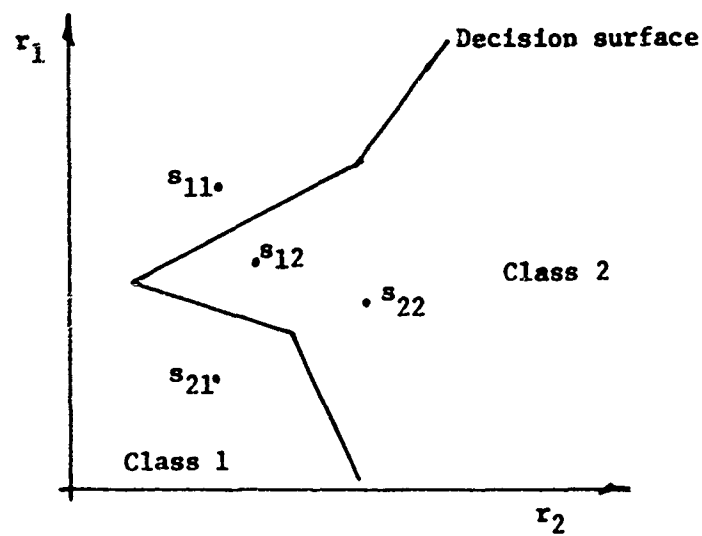


Figure 2.2. The Nearest Neighbor Rule Decision Surface

The distance-weighted k-nearest neighbor classifier assigns an unknown vector, R , to the class most heavily represented among its k nearest neighbors in the authority files. The 'representation' is computed as the sum of weights assigned to the k vectors. The weights, in turn, depend inversely upon the distance between the unknown and authority vectors. Using the notation defined above we may write:

$$w_{pq} = f(e_{pq}) \quad (2-74)$$

where w_{pq} is the weight assigned to the p -th vector in the authority file of class q .

The actual weight function depends upon the type of feature vector used, and will in turn affect the decision surface shape. $f(e_{pq})$ is usually defined such that:

$$\lim_{e_{pq} \rightarrow 0} w_{pq} = \lim_{e_{pq} \rightarrow 0} f(e_{pq}) = 1 \quad (2-75)$$

$$\lim_{e_{pq} \rightarrow \infty} w_{pq} = \lim_{e_{pq} \rightarrow \infty} f(e_{pq}) = 0 \quad (2-76)$$

The class weights, W_q , are then determined by:

$$W_q = \sum_{p=1}^P w_{pq} \quad (2-77)$$

w_{pq} is computed only for the k nearest neighbors of the unknown vector R , with all other w_{pq} set to zero. k may be a fixed number, in which

case the lowest k e_{pq} are selected to compute w_{pq} , or k may be variable and depend on the number of error terms below some limit:

$$w_{pq} = 0 \text{ if } e_{pq} > e_{\max}; 1 \leq p \leq P; 1 \leq q \leq Q. \quad (2-78)$$

In any case, there will exist a largest W_q

$$W_n \geq W_q \text{ for all } q. \quad 1 \leq q \leq Q \quad (2-79)$$

and R is assigned to class n . The decision surface defined by the distance-weighted k -nearest neighbor rule is decidedly non-linear.

CHAPTER III

FACIAL RECOGNITION AND AUTOMATIC TRAINING

3.1 Facial Profile Recognition

Once a training algorithm to select the vectors for the authority files has been devised and verified and the various system parameters optimized, the pattern recognition system is complete and the testing of its ability to recognize facial profiles may begin. Five experiments were conducted. The procedure in each was to divide the 120 facial profiles into two distinct sets, a training set and a test set. The activity sort training rule (to be described later in this chapter) was used to select vectors from the training set for the authority files. The recognition accuracy of the system was then tested with both the training and test sets. The recognition accuracy obtained on the test set is, of course, the only valid measure of the system's performance, since the test set consists solely of images that are unknown to the computer. The recognition accuracy on the training set, some of whose image feature vectors will be stored in the authority files, was included simply as a benchmark. In the remainder of this thesis whenever recognition accuracy is mentioned, the recognition accuracy on the independent test set is meant unless otherwise specified.

In experiments 1 - 4 training with the activity sort rule was terminated after 250 random selections from the training set had been made. In experiment 5, 500 selections were used because of the larger training set size. This was deemed a sufficient number of inputs to achieve maximum recognition accuracy based on the training times obtained in the tests of the activity sort training rule described in Section 3.4.

In experiment 1, samples 1-6 of the facial profile images in each class were used as the training set, while samples 7-12 of the facial profile images in each class were used as the test set. In experiment 2, this was reversed with samples 7-12 used as the training set and samples 1-6 as the test set. The results of these two experiments are shown in Tables 3.1 and 3.2, respectively. Note that the recognition accuracy on the test set is considerably higher in the first experiment. This is most likely due to the variations introduced in samples 1-6 of the facial profile images by changes in lighting, camera aperture, and subject position as described in Section 2.3. When used as a training set these images seem to provide authority files that better define the region in the feature vector space in which each class falls (i.e., a better decision surface) than do samples 7-12, where variations from image to image were held as small as possible.

Authority File Depth	Recognition Accuracy			
	Training set		Test set	
	No. Correct	% Correct	No. Correct	% Correct
1	34	56.7	27	45.0
2	48	80.0	34	56.7
3	57	95.0	46	76.7
4	59	98.3	43	71.7
5	60	100	41	68.3
6	60	100	41	68.3

Training Set -- Samples 1,2,3,4,5,6

Test Set - Samples 7,8,9,10,11,12

Table 3.1 Facial Profile Recognition Experiment 1 Results

Authority File Depth	Recognition Accuracy			
	Training set		Test set	
	No. Correct	% Correct	No. Correct	% Correct
1	53	88.3	26	43.3
2	59	98.3	32	53.3
3	60	100	26	43.3
4	60	100	26	43.3
5	60	100	26	43.3
6	60	100	26	43.3

Training Set - Samples 7,8,9,10,11,12

Test Set - Samples 1,2,3,4,5,6

Table 3.2 Facial Profile Recognition Experiment 2 Results

Experiments 3 and 4 (Tables 3.3 and 3.4, respectively) used random selections of 6 images per class in the training set and the remaining images in the test set. Maximum recognition accuracies on the test set were better than those obtained in experiment 1. Experiment 5 used a random selection of 9 images per class in the training set and the remaining images in the test set. The results, in Table 3.5, show the highest recognition accuracy achieved on any test set. This seems to indicate that if the activity sort rule is presented with a large training set, it can determine the decision surfaces (authority file vectors) to provide better recognition accuracy on independent data than if it is presented with a smaller training set. "Seems" is used in the preceding sentence because the small size of the test set makes generalizing the results very risky, but the preceding statement does agree with the expected outcome from probability theory. As more samples of the classes are obtained, the sample distribution becomes better defined and the decision surfaces can therefore be adjusted (i.e., authority file vectors picked) to provide better separation between classes and hence better recognition accuracy.

Notice that in all cases the maximum recognition accuracy on the test set does not occur at maximum authority file depth, as might be expected. Plots of recognition accuracy vs. authority file depth are given in Figure 3.1. The maximum recognition accuracy seems to occur at about one-half the maximum depth. The reason for this behavior is not clear--the small sample size makes any generalization difficult.

Authority File Depth	Recognition Accuracy			
	Training Set		Test Set	
	No. Correct	% Correct	No. Correct	% Correct
1	29	48.3	21	35.0
2	51	85.0	48	80.0
3	54	90.0	46	76.7
4	59	98.3	43	71.7
5	58	96.7	44	73.3
6	60	100	46	76.7

Training Set - Random selection 1,6 samples per class

Test Set - Remaining 60 images, 6 per class

Table 3.3 Facial Profile Recognition Experiment 3 Results

Authority File Depth	Recognition Accuracy			
	Training Set		Test Set	
	No. Correct	% Correct	No. Correct	% Correct
1	34	56.7	28	46.7
2	53	88.3	38	63.3
3	57	95.0	43	71.7
4	59	98.3	47	78.3
5	60	100	45	75.0
6	60	100	45	75.0

Training Set - Random selection 2, 6 samples per class

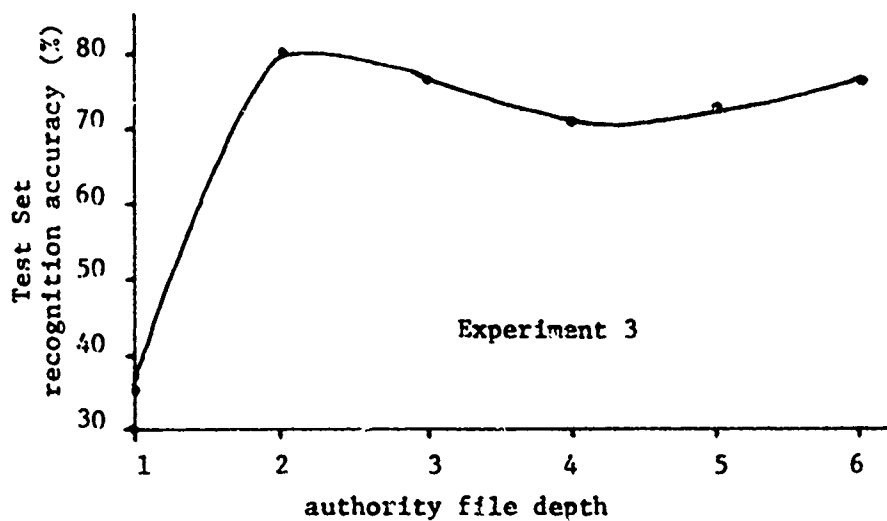
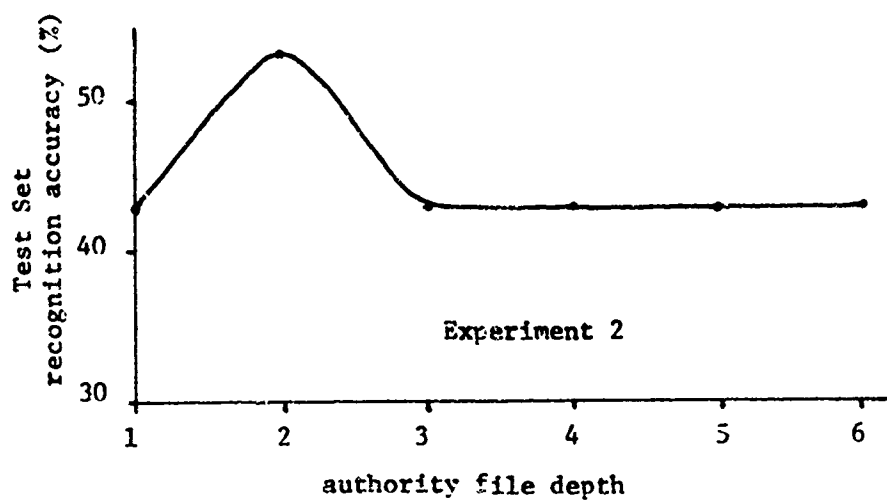
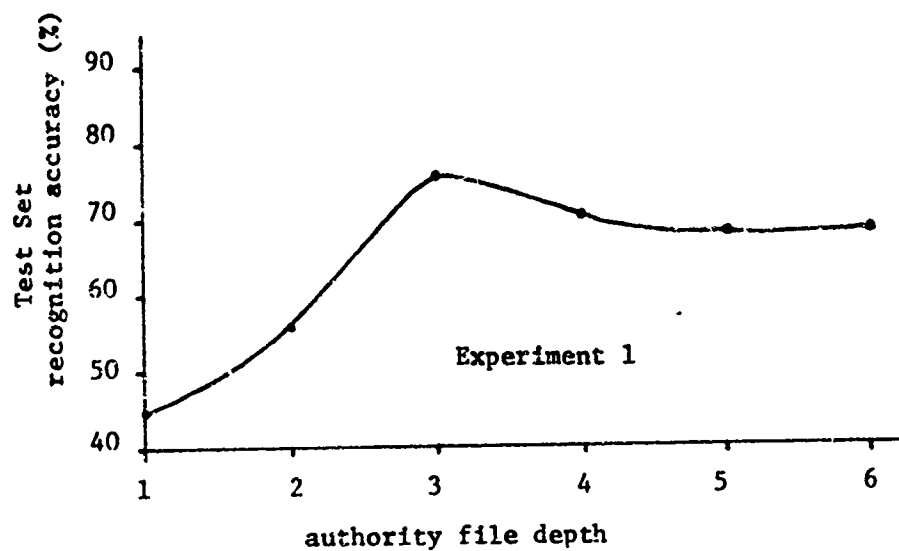
Test Set - Remaining 60 images, 6 per class

Table 3.4 Facial Profile Recognition Experiment 4 Results

Authority File Depth	Recognition Accuracy			
	Training Set		Test Set	
	No. Correct	% Correct	No. Correct	% Correct
1	62	68.9	21	70.0
2	73	81.1	22	73.3
3	84	93.3	25	83.3
4	83	92.2	27	90.0
5	89	98.9	23	76.7
6	88	97.8	22	73.3
7	88	97.8	25	83.3
8	90	100	23	76.7
9	90	100	26	86.7

Training Set - Random selection 2, 9 samples per class
 Test Set - Remaining 30 images, 3 per class

Table 3.5 Facial Profile Recognition Experiment 5 Results



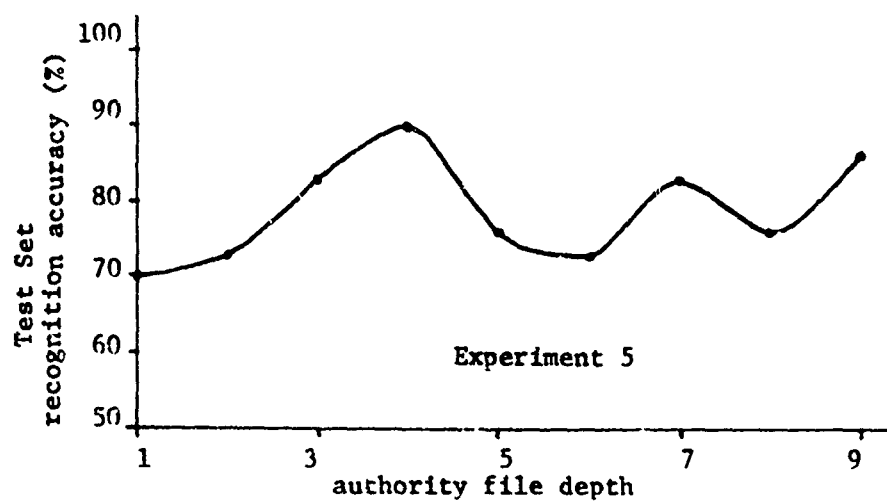
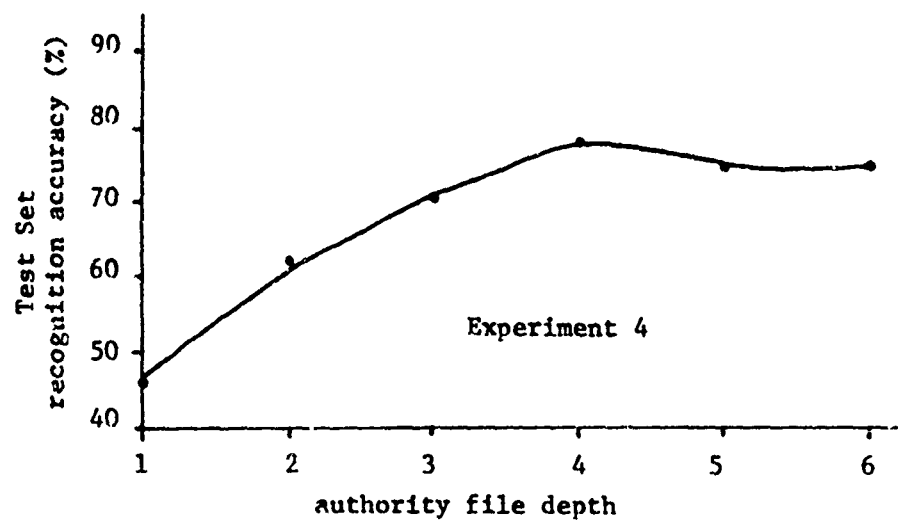


Figure 3.1 Recognition Accuracy vs. Authority File Depth

It was initially felt that the results discussed above did not have sufficiently high recognition accuracy, and because of the excellent recognition accuracy obtained by Dudani with a moment invariants feature extractor, it was decided to replace the circular autocorrelation function with moment invariant functions. The six moment invariant functions given in Section 2.5 were used twice, once on the image boundary and once on the silhouette, to obtain a 12-dimensional feature vector. The classification algorithm was modified to use the weight function and the feature vector normalization used by Dudani's aircraft recognition classifier. The last three facial profile recognition experiments were then repeated with no other changes. Typical feature vectors are given in Table 3.6.

The results are given in Tables 3.7 - 3.9. The low recognition accuracies on the test sets were totally unexpected. Recognition accuracies were, in all but one case, lower than those obtained using circular autocorrelation. There may be two reasons for this result. The class-to-class shape variations are often more subtle for facial profiles than for aircraft silhouettes, and due to the idiosyncracies of the television camera used for input to the computer, facial profiles exhibited more variation from sample to sample within a given class than the aircraft images. Under these conditions circular autocorrelation seemed to provide feature vectors with better separability between classes than the moment invariants did.

Table 3.6 Typical Moment Invariant Feature Vectors

Class	Feature Vector					
	1	2	3	4	5	6
1	.556	$.555 \times 10^{-1}$	$.186 \times 10^{-1}$	$-.577 \times 10^{-3}$	$.984 \times 10^{-2}$	$-.159 \times 10^{-3}$
2	.622	$.818 \times 10^{-1}$	$.339 \times 10^{-3}$	$.148 \times 10^{-5}$	$-.267 \times 10^{-3}$	$.100 \times 10^{-5}$
3	.468	.758	.220	$.624 \times 10^{-1}$	-.113	$-.647 \times 10^{-1}$
4	.672	.137	$.701 \times 10^{-2}$	$-.384 \times 10^{-4}$	$-.171 \times 10^{-2}$	$.214 \times 10^{-3}$
5	.641	$.629 \times 10^{-1}$	$.800 \times 10^{-1}$	$.151 \times 10^{-2}$	$.638 \times 10^{-1}$	$.547 \times 10^{-2}$
6	.618	.129	$.296 \times 10^{-1}$	$-.151 \times 10^{-2}$	$.932 \times 10^{-2}$	$-.103 \times 10^{-2}$
7	.519	$.595 \times 10^{-1}$	$.133 \times 10^{-1}$	$.124 \times 10^{-3}$	$-.194 \times 10^{-2}$	$-.350 \times 10^{-3}$
8	.505	.477	.190	$.540 \times 10^{-1}$	$-.326 \times 10^{-1}$	$-.128 \times 10^{-1}$
9	.635	.132	$.420 \times 10^{-3}$	$-.937 \times 10^{-6}$	$-.125 \times 10^{-3}$	$.299 \times 10^{-5}$
10	.548	.162	$.256 \times 10^{-1}$	$.150 \times 10^{-2}$	$.124 \times 10^{-1}$	$.678 \times 10^{-3}$

Class	Feature Vector					
	7	8	9	10	11	12
1	.707	$.876 \times 10^{-1}$	$.765 \times 10^{-2}$	$.289 \times 10^{-4}$	$.342 \times 10^{-2}$	$-.196 \times 10^{-3}$
2	.780	.102	$.291 \times 10^{-1}$	$.146 \times 10^{-2}$	$.128 \times 10^{-1}$	$.625 \times 10^{-3}$
3	.718	.401	.218	$.285 \times 10^{-1}$.182	$.582 \times 10^{-1}$
4	.762	.219	$.368 \times 10^{-1}$	$.124 \times 10^{-2}$	$-.217 \times 10^{-3}$	$.306 \times 10^{-2}$
5	.730	.107	$.197 \times 10^{-1}$	$.735 \times 10^{-3}$	$.568 \times 10^{-2}$	$-.542 \times 10^{-3}$
6	.772	.157	$.489 \times 10^{-1}$	$.429 \times 10^{-2}$	$.319 \times 10^{-1}$	$.128 \times 10^{-3}$
7	.721	$.422 \times 10^{-1}$	$.282 \times 10^{-2}$	$-.124 \times 10^{-4}$	$-.180 \times 10^{-2}$	$-.283 \times 10^{-4}$
8	.645	.392	$.675 \times 10^{-1}$	$.949 \times 10^{-2}$	$.480 \times 10^{-1}$	$-.242 \times 10^{-2}$
9	.766	.163	$.274 \times 10^{-1}$	$.954 \times 10^{-3}$	$-.988 \times 10^{-3}$	$-.157 \times 10^{-2}$
10	.737	$.646 \times 10^{-1}$	$.835 \times 10^{-2}$	$.124 \times 10^{-4}$	$-.933 \times 10^{-3}$	$.193 \times 10^{-3}$

Table 3.7 Moment Invariants Recognition Experiment 1 Results

Authority File Depth	Recognition Accuracy			
	Training Set		Test Set	
	No. Correct	% Correct	No. Correct	% Correct
1	33	55.0	22	36.7
2	40	66.7	23	38.3
3	47	78.3	36	60.0
4	56	93.3	29	48.3
5	60	100	29	48.3
6	60	100	32	53.3

Training Set - Random selection 1, 6 samples per class

Test Set - Remaining 60 images, 6 per class

Table 3.8 Moment Invariants Recognition Experiment 2 Results

Authority File Depth	Recognition Accuracy			
	Training Set		Test Set	
	No. Correct	% Correct	No. Correct	% Correct
1	37	61.7	25	41.7
2	46	76.7	30	50.0
3	56	93.3	26	43.3
4	58	96.7	33	55.0
5	60	100	27	45.0
6	60	100	31	51.7

Training Set - Random selection 2, 6 samples per class

Test Set - Remaining 60 images, 6 per class

Authority File Depth	Recognition Accuracy			
	Training Set		Test Set	
	No. Correct	% Correct	No. Correct	% Correct
1	36	40.0	10	33.3
2	56	62.2	14	46.7
3	68	75.6	16	53.3
4	74	82.2	15	50.0
5	82	91.1	21	70.0
6	87	96.7	19	63.3
7	90	100	19	63.3
8	90	100	17	56.7
9	90	100	19	63.3

Training Set - Random Selection 2, 9 samples per class
 Test Set - Remaining 30 images, 3 per class

Table 3.9 Moment Invariants Recognition Experiment 3 Results

In order to determine whether the results of the above experiments were acceptable, it was decided to compare the recognition accuracy of computer facial profile recognition against that of humans presented with the same data. Since humans are generally regarded as good pattern recognizers, it was felt that if the machine performed comparably to a human the pattern recognition system would be acceptable. Accordingly, the following experiment was devised. A facial profile photograph of each of the ten subjects was taken. These photographs became the human recognizer's "authority file." Three people were chosen for this experiment, the first two technically oriented and close to this work, the third non-technically oriented and unfamiliar with this work. Each person was given the set of reference photographs and seated in front of the Tektronix 611 display. The facial profiles after edge extraction, as presented in Appendix B, were selected at random and displayed on the CRT one at a time. The recognizer was given as long as he wished to make a classification by comparing the CRT image against the photographs. All 120 facial profiles were used, but because of limited disk storage samples 1-6 (on all classes) were presented first, and then samples 7-12.

The recognition accuracies of the three human recognizers are given in Table 3.10. It can be seen that the recognition accuracy on samples 7-12 is higher than the recognition accuracy on samples 1-6. There are two possible reasons for this; the recognizers may have had some difficulty in correlating the CRT images to the reference photographs, i.e., some early misclassifications were due to learning, and the larger

sample-to-sample variation within a given class for samples 1-6 may have reduced the human recognizers' accuracy. In any case, comparing these results with the computer results, it can be seen that using circular autocorrelation the computer at least matched the human recognition accuracy in all but experiment 2, while the results of experiment 5 indicate that with a sufficiently large training set it is possible for a computer pattern recognition system to achieve significantly higher recognition accuracy than a human on this problem. The best recognition accuracy obtained with moment invariants (70%) is comparable to human recognition accuracy on the 120 images (68-71%), a result similar to, although not quite as good as, that obtained by Dudani on aircraft recognition [5].

Subject	Recognition Accuracy					
	Samples 1-6		Samples 7-12		Samples 1-12	
	No. Correct	% Correct	No. Correct	% Correct	No. Correct	% Correct
1	39	65.0	46	76.7	85	70.8
2	38	63.3	44	73.3	82	68.3
3	38	63.3	46	76.7	84	70.0

Table 3.10 Human Facial Profile Recognition Accuracy

3.2 Automatic Training

In the following chapter, the selection and parameter optimization of two deterministic feature vector classification schemes will be discussed, while in this chapter the problem of building the authority file for the classifier will be addressed. It is the authority files' contents and depth (number of vectors) that determines the decision surfaces. If the feature vectors are well defined, or a sufficiently large sample size exists, the authority file vectors and depth for optimal recognition accuracy may be determined by computation (indeed, the authority file concept is not even necessary; one may write a set of equations describing the decision surfaces). However, outside of a theoretical analysis, this is seldom the case. The problem then is how, with the data available, are the authority files to be generated? The construction of the authority file is referred to as training. Duda and Fossum [44] have described an algorithm that computes an authority file vector for linearly separable data, i.e., the classes in the feature vector space may be separated by hyperplanes. Feature vectors generated by present feature extractors, however, tend not to fall in such neat classes, and in general a fairly convolved decision surface is required to separate the classes. Such a convolved decision surface may be obtained by using authority file depths greater than one. The authority files may be filled with vectors selected from some training set, i.e., feature vectors generated from a typical set of images to be recognized. The selection criterion is to obtain the

set of vectors that give the greatest recognition accuracy. The feature vectors selected to fill the authority files then determine the decision surfaces.

For a practical pattern recognition system operating in a changing environment, some sort of adaptive decision surface is desirable. This suggests a classification routine that is able to adjust its authority files to obtain a more nearly optimum set of decision surfaces. Such a routine should also be able to add new classes if necessary. Each classification attempt then, by such a pattern recognition system, would have the potential of changing the authority files (and hence decision surfaces) to adjust to new data. Two algorithms to provide this automatic training are discussed in the next section.

3.3 Two Training Algorithms

Consider a classifier of the nearest neighbor or distance weighted k-nearest neighbor type with authority files of depth n . A random feature vector is applied to the classifier and a classification made. If the classification is correct, the closest (Euclidian distance) vector in the authority files to the unknown vector that is also in the same class as the unknown vector is moved to the top of its authority file. If the classification is not correct, the unknown vector is 'pushed' on top of the appropriate authority file. All other vectors in this file are pushed down, and if the file is full, the bottom vector will be lost.

Under these rules the most 'important' vectors, i.e., those used most often in identification and that define 'critical' areas of the decision surfaces, will 'bubble' to the top of the authority files, while the vectors seldom used in identification will 'sink' to the bottom of the file. When a vector is incorrectly classified, its insertion into the authority file redefines the decision surface so that the misclassification will not occur again. If the authority file is full, it is the least used vector that will be lost when a new vector is added because of the bubble action described above.

A variation on this technique is to associate a counter with each vector in the authority files. Again, a random feature vector is applied to the classifier and a classification made. If the classification is correct, the closest feature vector in the authority files to the unknown vector that is also in the same class as the unknown vector has its counter incremented. If the classification is not correct, the vector in the appropriate authority file with the lowest number in its counter is replaced by the unknown vector and the counter is reset.

The effect of this rule is similar to that of the bubble sort rule described above. Authority file vectors with the highest 'activity,' i.e., those used most often in classification, are retained, while seldom used vectors are removed so that a new, and perhaps more important vector may be added. These two authority file sorting rules do require a 'teacher,' since the classification of the 'unknown' vector

must be known to be correct or not, and if incorrect the class of the 'unknown' vector must be known so that the correct authority file may be modified. These requirements are not restrictive if these rules are used when the authority files are first filled from a training set, but if the authority files are to be modified when the pattern recognition system is operating, some sort of feedback as to the correctness of the classification must be provided. The first application, that of initial training, is investigated in the following sections, while the second application of these sorting rules, that of decision surface modification during operation, is left for future research.

3.4 An Experimental Comparison

A simple experiment was performed to verify the expected performance of the two authority file sorting rules. The circular autocorrelation feature extractor and the distance-weighted k-nearest neighbor classifier were used. Authority file depth was set to three. The training and testing set was samples 1-6 of the facial profile images for each of the ten classes. In the first test, the authority files' contents were adjusted manually by cut-and-try method to obtain the best possible recognition accuracy. Next the bubble sort technique was run for 500 random selections from the training set and then tested. Then the activity sort rule was run for 500 random selections and tested. The results are shown in Table 3.11. The samples to train column indicates the point (in number of random inputs) after which the authority files began to 'oscillate.' Since some authority files were

not deep enough to hold all the vectors necessary for 100% recognition accuracy on that class, the lower importance vectors would swap or oscillate in and out of the files. At this point, no further increase in recognition accuracy could be obtained.

Test No.	Sort Type	Authority File Depth	Samples to Train	Recognition Accuracy		
				No. Correct	% Correct	No. classes 100% correct
1	manual	3	--	49	81.7	7
2	bubble	3	111	48	80.0	6
3	activity	3	111	52	86.7	6
4	activity	4	179	59	93.3	9
5	activity	2	181	43	71.7	3

Table 3.11 Sort Algorithm Recognition Accuracies

From Table 3.11 it can be seen that the manual and bubble sort produced about the same recognition accuracy, while that of the activity sort was slightly higher. One reason that the activity sort performed better than the bubble sort may be because of the inherent integration provided by the activity sort counter. Consider, for example, an authority file which contains a vector that is often used (i.e., is the closest to the input vector) in classification. Suppose now an input sequence occurs in which the other vectors in the authority file are used and hence bubble to the top of the file, or several misclassified vectors are inserted into the file. If this occurs the

important (often used) vector may end up at the bottom of the authority file and be pushed out by the next vector inserted. Since the input sequence is random, such a sequence is most likely to occur for small file depths. With the activity sort an often used vector will build up a count significantly larger than less important vectors, and an input sequence that did not use this vector would have to be long enough to build up the counts of the other feature vectors to values greater than that of the 'important' vector before it would be replaced. Since a longer input sequence of this type is required, it is less likely to occur and the activity sort files remain more stable.

The automatic training rules were not able to match the manual selection of feature vectors for the number of classes 100% correct, which is not surprising since the automatic routines were designed to optimize the total recognition accuracy without regard to class. Table 3.12 shows the recognition accuracy of the three methods for each class. Included in Tables 3.11 and 3.12 are two training attempts with the activity sort rule for authority file depths of 2 and 4. The recognition accuracy behaved as expected, lower for a depth of 2 and higher for a depth of 4. Notice that the number of inputs required to train the authority files increases for depths of 2 and 4. For a file depth of 2 or 4 and a random input selection with a uniform probability density function, an average of 3 passes through the input set is required before the activity file contents start to oscillate.

Test No.	Recognition Accuracy, % Correct/Class									
	1	2	3	4	5	6	7	8	9	10
1	50	100	100	100	100	50	1.7	100	100	100
2	50	100	100	83	100	67	0	100	100	100
3	50	100	100	83	100	50	83	100	100	100
4	83	100	100	100	100	100	100	100	100	100
5	33	83	100	83	67	50	50	100	50	100

Table 3.12. Sort Algorithm Recognition Accuracies per Class

The activity sort rule has a feature which may be useful if it is to be used to maintain the authority files in an operating pattern recognition system. It can be seen that under such conditions the counts associated with some vectors may become very large. The activity sort rule may be modified to divide the contents of all counters for a given authority file by a constant when any count in that file exceeds a preset value. This technique tends to give the authority files a certain 'forgetfulness.' Any authority file vector that is not the closest vector to the unknown (i.e., that does not have its counter incremented) a certain number of times for some number of classifications (depending on the values of the constants) will have its count reduced to zero by the divisions and will be erased when a new vector is entered. Because of its high recognition accuracy and the potential for implementing this limited memory time feature, the activity sort rule was chosen to fill the authority files in the remaining work.

3.5 A Large Sample Size Experiment

Although the activity sort automatic training algorithm seemed to perform well in the experiments described in the preceding section, there were two major flaws in these experiments. The input sample size was really too small to determine conclusively how well the training algorithms worked, and the test samples were the same as the training samples rather than independent data. It was decided to undertake an experiment using the data and pattern recognition routines of Dudani [5], with the activity sort rule for authority file construction, to obtain a more meaningful measure of the algorithm's performance.

Dudani addressed the problem of automatic aircraft identification. He defined a six-class problem and used a moment invariant feature extractor. The six authority files consisted of 551 12-tuples each. The authority files were obtained from images of the aircraft at roll angles from 0 to 90° and azimuth angles from -70 to 70°, both in 5-degree increments. The total number of vectors in the authority files was thus 3306. A distance-weighted k-nearest neighbor classifier was used. Dudani's test set consisted of 132 images obtained independently and in addition to the 3306 images used in the authority files. Of the 132 test images, 22 images were from each of the six classes, with random roll and elevation angles. The performance of the original system is shown on the first line of Table 3.13. The last column lists the image numbers from the test set that were incorrectly classified.

Authority File Depth	Training Samples	Recognition Accuracy		Incorrectly Classified Test Image Numbers
		No. Correct	% Correct	
551	----	126	95.4	4,23,31,83,125,130
184	3000	116	87.9	4,5,16,18,23,31,41, 61,62,118,122,125, 128,129,130,131
184	6000	121	91.7	16,23,31,64,75,90, 125,127,128,130,131

Table 3.13 Activity Sort Rule on a Large Training Set

To test the activity sort training rule, the authority file depths of Dudani's classifier were arbitrarily reduced to one-third of their original size, from 551 to 184. The activity sort rule was used to pick the new authority file contents from the training set consisting of the 3306 vectors which comprised the original authority files. After 3000 random samples from the training set had been examined by the training algorithm, the recognition accuracy was tested with Dudani's 132 independent test images. The training then continued for another 3000 random samples from the training set and the recognition accuracy was again tested with the 132 test images. The results are shown above in Table 3.13. It can be seen that after 6000 samples the recognition accuracy was approaching that obtained with the 551-deep authority files (92% versus 95%). It is interesting to note that test image numbers 23, 31, 125, and 130 are missed in all cases; this might

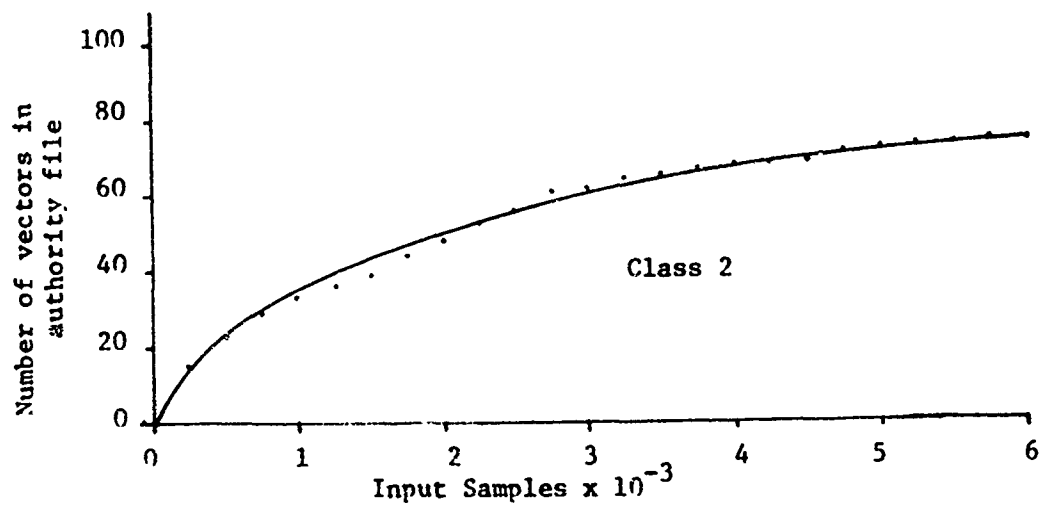
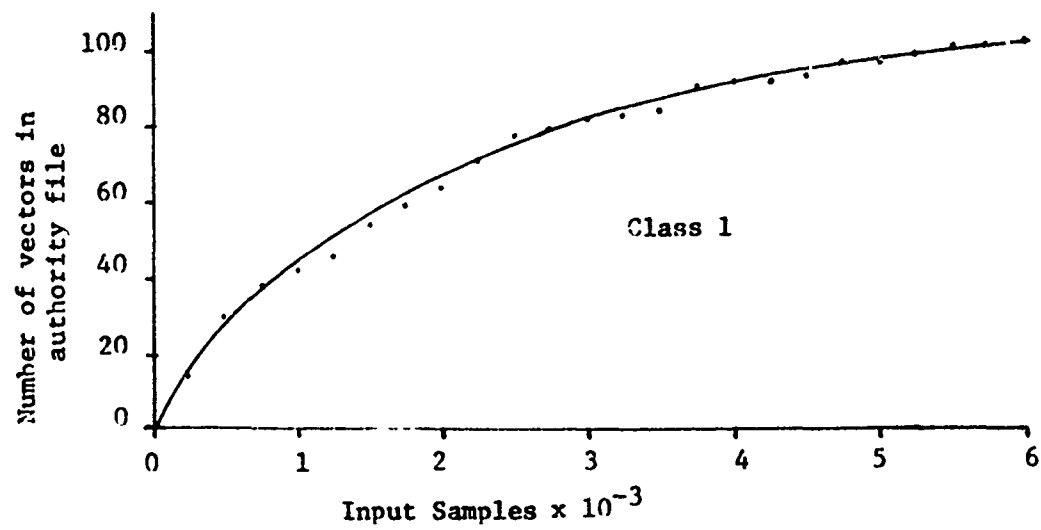
indicate the presence of inaccurate training or test data. In Table 3.14 the number of vectors in each authority file and the number of samples in each class presented to the training algorithm during training is given at 250-sample increments. It can be seen that the number of samples of each class presented to the training algorithm remained fairly equal. The number of vectors in the authority files versus the total number of input samples during training are graphed in Figure 3.2. The authority files seem to have filled at a logarithmic rate. Note that the authority files were not full when the experiment was terminated and that the number of vectors in each file varies considerably from class to class.

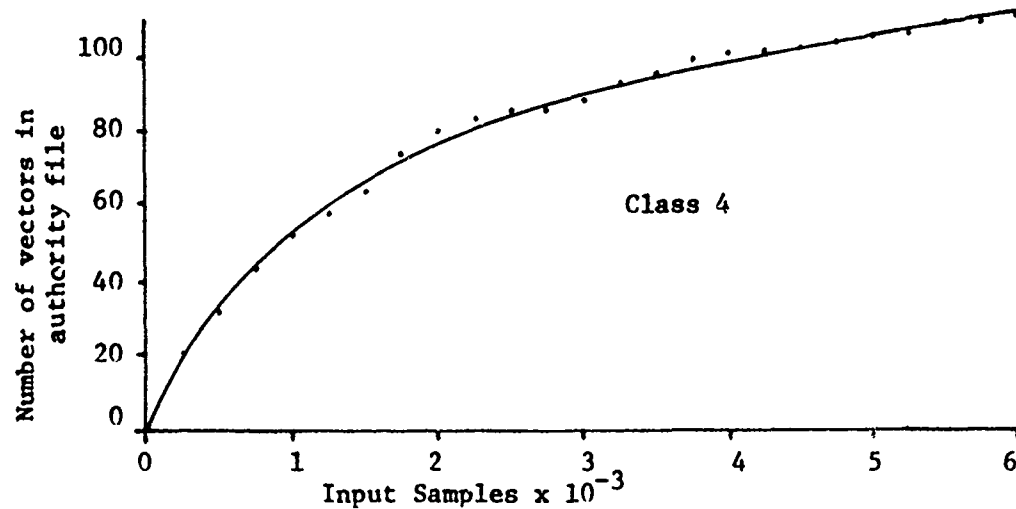
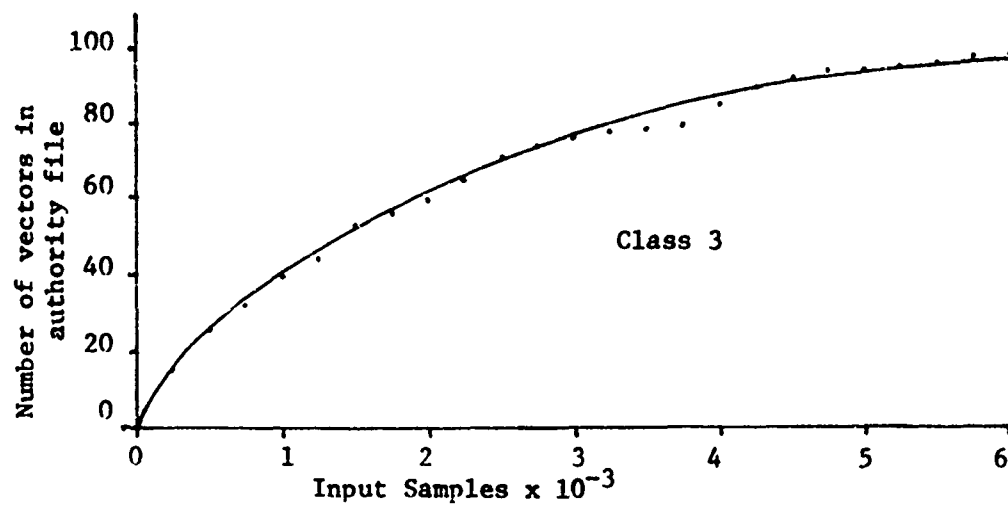
The experiment was terminated after 6000 training samples because of the computation time involved (60 hours on the PDP-9 computer). It would have been desirable to continue until the authority files were full, or to reduce the depth of the authority files (to, say, 50) to determine exactly which vectors would be saved and the recognition accuracy then achieved. This experiment does show, however, that the vectors selected by the activity sort rule will provide good recognition accuracy when tested on an independent test set.

Training Samples	No. of Vectors in Authority Files						No. of Samples Presented					
	1	2	3	4	5	6	1	2	3	4	5	6
	F-4	Mirage IIIC	MIG 21	F-105	F-104	B-57	F-4	Mirage IIIC	MIG 21	F-105	F-104	B-57
250	15	15	17	21	26	15	33	42	36	50	49	40
500	30	22	27	32	38	19	85	81	73	94	89	78
750	39	29	33	44	46	25	123	119	117	143	128	120
1000	43	33	40	53	50	28	158	156	157	187	177	165
1250	47	36	46	58	56	30	199	203	202	227	216	203
1500	54	39	54	64	61	31	251	252	252	261	247	237
1750	60	44	57	74	63	33	290	293	289	318	279	281
2000	66	48	60	80	69	34	341	342	334	356	317	310
2250	72	52	67	83	74	35	379	393	372	405	356	345
2500	78	56	71	86	80	39	432	430	418	441	398	381
2750	80	60	74	86	91	42	478	470	450	480	439	423
3000	83	61	77	88	93	42	522	516	504	523	481	454
3250	84	64	78	93	98	45	565	560	550	569	514	492
3500	85	65	79	95	99	46	605	596	588	609	562	540

Training Samples	No. of Vectors in Authority Files						No. of Samples Presented					
	1	2	3	4	5	6	1	2	3	4	5	6
	F-4	Mirage IIIC	MIC 21	F-105	F-104	B-57	F-4	Mirage IIIC	MIC 21	F-105	F-104	B-57
3750	91	67	80	99	102	47	639	642	631	654	612	575
4000	93	68	86	100	102	48	685	680	676	695	645	619
4250	93	68	90	101	102	49	724	717	719	739	688	663
4500	95	69	93	102	104	49	762	758	769	773	729	709
4750	98	71	95	103	105	50	798	789	814	810	777	762
5000	98	72	95	105	107	50	832	841	858	851	815	803
5250	100	72	96	106	108	50	872	885	904	895	853	841
5500	102	72	97	109	108	51	908	924	950	936	890	892
5750	102	73	98	109	108	52	944	965	994	982	924	941
6000	103	73	99	111	110	52	996	1001	1034	1016	971	982

Table 3.14 Training Algorithm Statistics





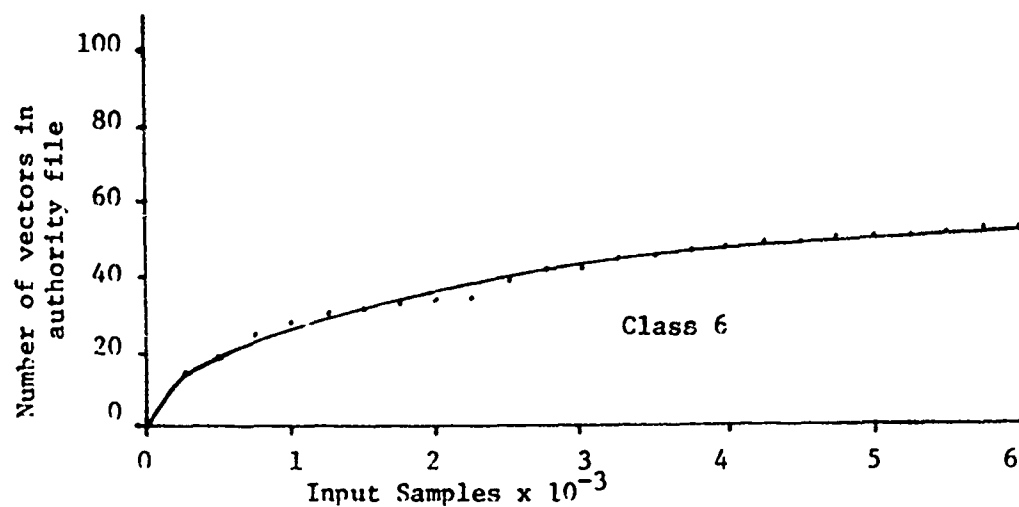
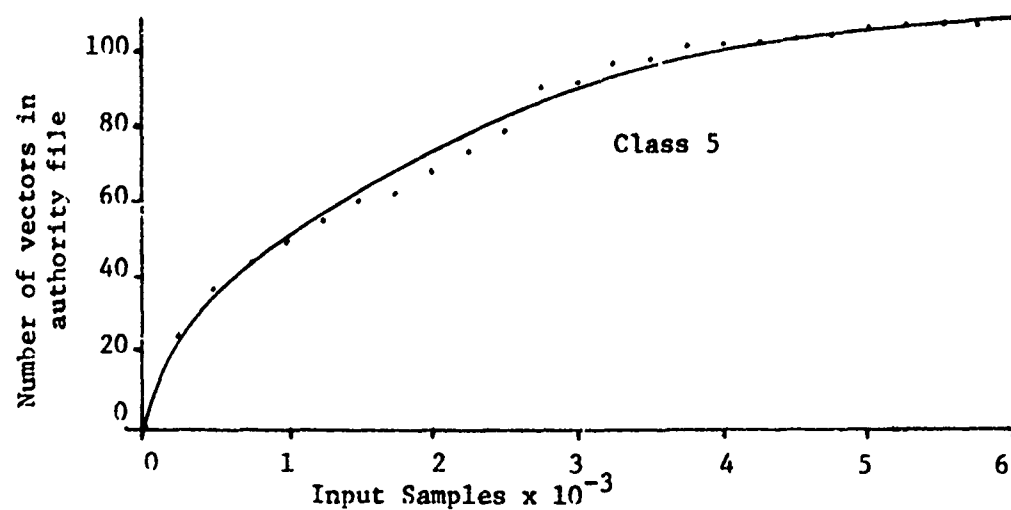


Figure 3.2 Number of Vectors in Authority Files vs. Input Samples
(Class 1 through Class 6)

CHAPTER IV

SYSTEM OPTIMIZATION

4.1 Introduction

This chapter concerns itself with the optimization of the circular autocorrelation feature extractor, a comparison of the nearest neighbor classifier and the distance-weighted k-nearest neighbor classifier, and the need for normalization of the feature vector. A demonstration of the translation and size invariance of the circular autocorrelation function is provided. The behavior of the circular autocorrelation function under rotation is also demonstrated. The circular autocorrelation function performance for various values of parameters a_m , M , and N is investigated. The 120 facial profile feature vectors generated with the chosen a_m , M , and N parameters are listed.

Two distance-weighted k-nearest neighbor classifier weight functions are described, one dependent upon the Euclidian distance between the unknown feature vector and the authority file vector, e_{pq} , and the other dependent upon $(e_{pq})^2$. Recognition accuracy of a classifier using each function is obtained. The recognition accuracy of a nearest neighbor classifier is compared to that of a distance-weighted k-nearest neighbor classifier. Normalization of the feature vectors before classification is discussed, and the mean and standard

deviation of the 12 circular autocorrelation feature vector components over the 120 facial profile samples are computed.

Although the feature extractor and classifier are discussed in separate sections, the development and optimization of both occurred simultaneously. It should also be pointed out here that the two algorithms interact to some extent, making optimization a very difficult task.

4.2 Circular Autocorrelation

After the circular autocorrelation function feature extractor was written, it was felt that an experimental demonstration of size invariance and behavior under image rotation was in order. It was decided to arbitrarily set the circular autocorrelation parameters at:

$$M = 1$$

$$N = 12$$

$$a_m = 1/2$$

so that

$$u = \frac{\sqrt{A}}{2} \cos\left(\frac{\pi(n-1)}{12}\right) \quad (4-1)$$

$$v = \frac{\sqrt{A}}{2} \sin\left(\frac{\pi(n-1)}{12}\right) . \quad (4-2)$$

The area A was computed as:

$$A = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m\gamma, n\gamma) \quad (4-3)$$

that is, the sum of the one cells. The actual $g(1,n)$ output was multiplied by 100 and truncated. It was felt that this would be sufficiently accurate and allowed integer arithmetic to be used in the classifier. This may be written as:

$$g(1,n) = \left[\frac{100}{A} g(u,v) \right] . \quad (4-4)$$

A simple experiment was performed to verify the theory and also to obtain an idea of the amount of feature vector variation to be expected for real images. A white rectangle 9" x 12" was placed in front of the television camera and the feature vector for various rotation angles was computed. The results are shown in Table 4.1 as samples number 1-17. The numbers in the area column are the total number of one cells in each array. The feature vector of sample number 1 was used as the authority file vector. Since a rotation of the input image by π/N radians (in this case $\pi/12$ radians or 15°) causes the circular autocorrelation feature vector components to rotate (see Section 2.5 and Eqs. 2-50 and 2-51); each unknown feature vector must be compared against the 12 possible variations of the authority file vector to find the best match. The numbers in the closest match, angle, and error² columns refer to the $\pi/12$ radian rotation (angle) for which the smallest Euclidian distance squared (error²) between the unknown and sample number 1 was obtained. Because of the rectangle's symmetry, the circular autocorrelation function feature vector of the rectangle should also be symmetric,

that is:

$$g(1,n) = g(1, 14-n) \quad (4-5)$$

for $n = 2, 3, 4, 5$.

That this feature vector symmetry was not quite achieved points to distortion in the input system.

It can be seen from Table 4.1 that the error for images 3-6 is considerably larger than that for images at $\pi/12$ radian rotation increments. This is to be expected since the feature vectors for images rotated at other than $\pi/12$ increments will not, in general, directly correspond to the zero radian feature vector (see Section 2.5). The other errors are probably due to input distortion, and give an idea of the ultimate accuracy of the system. Size invariance was tested by moving the camera farther from the rectangle and obtaining two more images. These are given as samples numbered 18 and 19 in Table 4.1. The area of these two images was reduced by about a factor of 5 from the other images, producing images less than one-half the size of the comparison image. As can be seen from the table, there was little change in the feature vectors.

Once it was verified that the feature extractor was performing as expected, the next step was to optimize the values of a_m , M , and N . This was attempted with the following experiment. An input space of 60 facial profile images, 6 images per subject, for the 10 subjects was chosen. A classifier of the distance-weighted k -nearest neighbor type with k fixed at 10 and the $(e_{pq})^2$ dependent weight function (to be described in the next section) was used. The classifier

Sample No.	Area Rotation angle (°)	Feature Vector Components, n												Closest Match	
		1	2	3	4	5	6	7	8	9	10	11	12	angle	error
1	19323	56	50	45	41	39	39	42	40	39	41	44	49	-	-
2	19323	56	50	45	41	39	39	42	40	39	41	44	49	0	0
3	19102	54	52	46	42	40	39	40	40	39	41	43	48	0	17
4	19152	52	54	47	43	40	39	40	41	39	40	43	47	15	44
5	19172	51	55	48	44	41	39	39	42	39	40	42	46	15	17
6	19071	49	56	49	44	41	39	39	42	40	40	41	45	15	4
7	18959	49	55	50	45	41	39	39	41	40	40	41	45	15	3
8	19151	43	48	55	51	45	42	39	40	42	40	39	41	30	6
9	19149	40	43	49	55	50	45	42	40	40	42	40	39	45	6
10	19145	38	40	44	49	55	51	45	43	40	40	40	39	60	12
11	19138	38	33	40	44	49	56	51	46	43	40	40	41	75	15
12	19337	42	39	39	40	44	49	55	51	45	41	39	39	90	4
13	18871	39	41	40	40	41	44	49	55	50	45	41	39	105	3
14	18988	39	39	42	40	40	41	44	49	55	50	44	41	120	3
15	19039	41	39	39	42	40	40	41	44	49	54	49	44	135	7
16	18575	44	42	39	39	42	40	39	41	44	49	54	50	150	6
17	18547	49	45	41	39	40	42	39	40	42	44	49	56	165	5
18	3981	57	50	45	41	39	39	42	40	40	42	46	50	0	8
19	4183	54	52	45	43	39	39	41	41	40	43	43	48	0	21

Table 4.1 Circular Autocorrelation Feature Vectors under Image Rotation and Size Change, for 9" x 12" rectangle.

searched only the zero angle feature vector sequences, that is, rotational invariance was not attempted. The authority file depths were set to three, i.e., three feature vectors per class were used to define each decision surface, and the authority files were filled using the activity sort training algorithm (described in Chapter III). Six sets of values for a_m , M and N for the circular autocorrelation function were then compared by training the machine on the circular autocorrelation feature vectors derived from the 60 images and testing the recognition accuracy of the machine using the same 60 images. The results of this experiment are summarized in Tables 4.2 and 4.3. Typical feature vectors for parameter sets 1, 3, 4, and 5 are given in Table 4.4, while Table 4.5 contains the feature vectors for all 120 facial profile images for parameter set 2; $a_m = 1/2$, $M=1$, $N=12$.

From Table 4.3 it can be seen that subjects 1, 6, and 7 were consistently misclassified as belonging to another class for all parameter sets and subject 9 was misclassified for 4 of the 5 sets. There was no apparent pattern to the misclassifications. It seems that these particular images are in some way more 'variable' than the other images and the region in the feature vector space to which they belong is harder to define. The low recognition accuracy of parameter sets 1, 3, and 5 is not surprising when the vectors in Table 4.4 are examined. For parameter set 1 ($a_m = 1$) there are several zero terms in the feature vector, terms which contribute no information, while for parameter set 3 ($a_m = 1/4$) the terms of the feature vectors exhibit little variation from class to class. The recognition accuracies

Parameter Set	a_m	M	N	Recognition Accuracy		
				No. Correct	% Correct	No. of Classes 100% Correct
1	1	1	12	47	78.3	5
2	1/2	1	12	52	86.7	6
3	1/4	1	12	49	81.8	6
4	1/2, 1/√2	2	6	51	85.0	6
5	1/3, 1	2	6	50	83.3	5

Table 4.2 Recognition Accuracy for Selected Circular Autocorrelation Function Parameters.

Parameter Set	Recognition Accuracy, % Correct/Class									
	1	2	3	4	5	6	7	8	9	10
1	50	100	50	100	100	50	50	100	83	100
2	50	100	100	83	100	50	83	100	100	100
3	50	100	100	100	100	67	50	100	50	100
4	50	100	100	100	100	67	67	100	67	100
5	50	83	100	100	100	67	67	100	67	100

Table 4.3 Recognition Accuracy per Class for Selected Circular Autocorrelation Function Parameters.

Parameter Class Set		Feature Vector Component, n											
		1	2	3	4	5	6	(1) 7	(2) 8	(3) 9	(4) 10	(5) 11	(6) 12
1	1	0	0	0	0	10	31	35	18	5	0	0	0
3	1	48	51	52	58	64	68	70	69	66	57	50	49
4	1	6	16	45	58	37	13	(0)	(7)	(34)	(57)	(27)	(5)
5	1	31	39	57	64	56	35	(0)	(4)	(31)	(54)	(25)	(4)
1	2	0	0	0	0	8	27	40	24	8	0	0	0
3	2	44	44	48	56	62	67	70	68	63	55	47	43
4	2	4	8	38	59	35	11	(0)	(2)	(27)	(55)	(25)	(3)
5	2	28	33	54	63	53	32	(0)	(1)	(24)	(53)	(23)	(1)
1	3	0	0	0	1	23	20	13	7	5	0	0	0
3	3	38	41	47	56	62	64	63	59	53	47	38	36
4	3	7	15	42	41	28	11	(0)	(6)	(36)	(36)	(22)	(6)
5	3	23	35	54	53	41	25	(0)	(4)	(32)	(31)	(18)	(3)
1	4	0	0	0	0	9	19	30	28	15	0	0	0
3	4	40	41	44	49	54	60	63	67	65	57	49	43
4	4	4	9	30	50	40	10	(0)	(4)	(23)	(47)	(31)	(4)
5	4	23	28	44	56	56	32	(0)	(3)	(21)	(44)	(30)	(3)
1	5	0	0	0	0	13	32	34	17	4	0	0	0
3	5	44	47	53	61	67	68	66	64	58	51	47	45
4	5	7	16	46	54	28	13	(0)	(6)	(39)	(54)	(20)	(6)
5	5	29	40	60	59	48	34	(0)	(5)	(37)	(51)	(19)	(5)
1	6	0	0	0	1	13	31	33	20	6	0	0	0
3	6	38	43	47	57	63	67	65	63	58	51	42	41

Continued...

Para- meter Class Set		Feature Vector Component, n											
		1	2	3	4	5	6	(1) 7	(2) 8	(3) 9	(4) 10	(5) 11	(6) 12
4	6	4	10	39	52	27	9	(0)	(4)	(33)	(48)	(20)	(4)
5	6	21	34	53	59	46	28	(0)	(3)	(28)	(45)	(16)	(3)
1	7	0	0	0	0	5	24	40	29	8	0	0	0
3	7	49	51	54	59	65	70	71	71	68	59	54	51
4	7	6	12	39	64	41	15	(0)	(5)	(30)	(61)	(31)	(6)
5	7	34	38	58	67	60	38	(0)	(3)	(27)	(58)	(28)	(4)
1	8	0	0	0	0	13	22	20	19	13	0	0	0
3	8	44	50	53	53	55	54	52	54	54	50	49	48
4	8	11	20	40	35	20	15	(1)	(12)	(37)	(36)	(13)	(8)
5	8	31	40	47	44	42	37	(0)	(11)	(37)	(38)	(12)	(7)
1	9	0	0	0	1	12	25	34	24	10	0	0	0
3	9	40	44	48	54	58	64	66	67	60	53	45	42
4	9	3	10	29	52	32	10	(0)	(3)	(24)	(48)	(24)	(3)
5	9	24	34	47	58	49	31	(0)	(2)	(21)	(46)	(20)	(2)
1	10	0	0	0	1	4	14	32	30	13	1	0	0
3	10	42	43	44	48	55	60	61	62	59	53	49	45
4	10	9	15	31	49	40	19	(0)	(8)	(22)	(46)	(30)	(8)
5	10	28	31	46	54	51	38	(0)	(6)	(19)	(41)	(27)	(6)

Table 4.4 Typical Feature Vectors for Selected Circular Autocorrelation Function Parameters.

FEATURE VECTOR COMPONENT, R

CLASS	SAMPLE	NAME	1	2	3	4	5	6	7	8	9	10	11	12
1	1	J. KAUPER	6	10	16	28	45	57	58	49	37	22	13	12
1	2	J. KAUPER	6	8	14	21	35	50	59	51	41	23	14	10
1	3	J. KAUPER	4	5	12	20	33	51	60	53	35	17	10	6
1	4	J. KAUPER	4	5	11	18	31	49	60	54	40	23	12	6
1	5	J. KAUPER	7	8	15	22	35	53	60	53	36	23	15	9
1	6	J. KAUPER	6	7	13	23	41	55	57	48	36	20	11	8
1	7	J. KAUPER	5	6	12	22	37	53	59	50	36	19	11	8
1	8	J. KAUPER	5	8	14	23	35	54	61	53	37	23	13	9
1	9	J. KAUPER	6	7	15	23	36	54	61	52	37	22	14	9
1	10	J. KAUPER	5	7	13	22	34	49	59	49	37	21	13	8
1	11	J. KAUPER	6	8	14	21	33	49	58	50	37	21	14	8
1	12	J. KAUPER	6	8	14	23	34	49	59	51	38	25	15	9
2	1	G. PELOSO	4	4	8	20	38	53	59	51	35	20	11	6
2	2	G. PELOSO	2	3	6	16	36	52	59	50	33	18	9	5
2	3	G. PELOSO	2	2	6	18	36	52	61	51	35	19	9	5
2	4	G. PELOSO	6	3	14	27	43	59	63	54	36	23	14	9
2	5	G. PELOSO	7	6	13	26	43	58	63	53	38	23	14	8
2	6	G. PELOSO	4	5	10	21	38	54	62	53	38	21	11	7
2	7	G. PELOSO	3	3	8	18	36	52	62	55	41	22	12	6
2	8	G. PELOSO	2	4	8	17	35	51	63	56	40	22	11	6
2	9	G. PELOSO	2	4	8	18	33	52	62	56	40	23	12	6
2	10	G. PELOSO	2	3	8	19	34	53	63	56	39	23	12	6
2	11	G. PELOSO	2	3	7	16	34	50	61	53	39	20	10	6
2	12	G. PELOSO	3	4	8	19	36	53	62	55	41	22	12	7
3	1	C. RUCCILLA	7	9	15	28	42	53	62	55	28	19	11	8
3	2	C. RUCCILLA	3	5	12	24	43	53	61	34	27	14	8	4
3	3	C. RUCCILLA	5	5	13	25	41	50	60	37	26	15	9	5
3	4	C. RUCCILLA	3	6	11	28	46	53	52	36	25	14	7	4
3	5	C. RUCCILLA	2	5	11	25	42	53	51	36	22	13	5	3
3	6	C. RUCCILLA	2	5	11	24	39	52	44	31	21	11	6	4
3	7	C. RUCCILLA	2	6	13	26	41	53	54	43	30	17	8	4
3	8	C. RUCCILLA	5	8	10	29	43	59	64	50	35	22	12	6
3	9	C. RUCCILLA	4	6	16	27	44	58	56	41	27	17	9	5

CLASS	SAMPLE	NAME	FEATURE VECTOR COMPONENT, n															
3	10	C. BUCCILLA	4	6	14	28	44	58	57	43	31	18	8	5				
3	11	C. BUCCILLA	4	6	15	28	46	57	54	39	28	16	8	5				
3	12	C. BUCCILLA	3	6	14	26	45	57	56	40	27	15	7	4				
4	1	K. MIELKE	4	5	9	17	30	42	50	49	40	22	10	5				
4	2	K. MIELKE	5	7	10	10	31	44	52	51	42	24	11	7				
4	3	K. MIELKE	1	4	8	15	27	34	38	38	32	18	7	3				
4	4	K. MIELKE	1	3	6	15	27	35	41	40	33	18	5	3				
4	5	K. MIELKE	1	3	8	16	27	35	39	37	33	17	7	3				
4	6	K. MIELKE	6	8	13	22	35	45	47	40	32	18	9	7				
4	7	K. MIELKE	4	6	12	21	35	45	48	44	36	19	9	5				
4	8	K. MIELKE	5	7	12	20	32	43	46	44	37	19	9	6				
4	9	K. MIELKE	3	5	12	20	31	44	43	45	38	21	10	5				
4	10	K. MIELKE	6	7	15	23	34	46	51	43	37	24	12	7				
4	11	K. MIELKE	5	7	12	22	32	45	46	43	36	19	9	6				
4	12	K. MIELKE	2	5	8	16	28	40	48	40	38	19	8	4				
5	1	G. TAKASAKI	7	9	16	30	46	56	54	44	28	20	13	9				
5	2	G. TAKASAKI	7	10	16	29	47	58	57	48	33	21	12	9				
5	3	G. TAKASAKI	5	6	12	26	47	57	59	46	30	17	9	6				
5	4	G. TAKASAKI	4	7	12	26	44	57	57	46	29	18	10	6				
5	5	G. TAKASAKI	5	9	17	30	49	61	58	47	30	18	11	8				
5	6	G. TAKASAKI	3	6	11	22	41	58	61	49	30	16	9	5				
5	7	G. TAKASAKI	5	6	12	26	45	56	57	45	29	17	10	7				
5	8	G. TAKASAKI	5	8	15	28	47	59	58	46	29	17	11	7				
5	9	G. TAKASAKI	3	6	12	23	43	58	60	49	31	17	9	5				
5	10	G. TAKASAKI	5	6	12	25	47	56	56	43	30	16	9	6				
5	11	G. TAKASAKI	3	6	11	23	43	53	59	46	29	16	9	5				
5	12	G. TAKASAKI	4	6	11	26	44	59	60	48	29	17	9	6				
6	1	C. ROBINSON	4	6	10	23	39	50	52	42	27	16	9	5				
6	2	C. ROBINSON	4	7	14	25	39	51	55	45	33	21	11	6				
6	3	C. ROBINSON	4	7	15	25	40	51	54	43	32	18	11	6				
6	4	C. ROBINSON	2	5	9	20	35	48	55	46	32	18	9	4				
6	5	C. ROBINSON	6	8	13	27	41	54	52	43	30	19	11	8				
6	6	C. ROBINSON	6	9	16	31	48	57	53	42	30	20	11	7				
6	7	C. ROBINSON	2	5	11	20	34	51	57	47	33	19	11	6				

CLASS	SAMPLE	NAME	FEATURE VECTOR COMPONENT, n											
9	6	J. SWARTZ	4	7	12	20	34	45	52	46	34	18	11	7
9	7	J. SWARTZ	2	5	9	15	27	42	54	52	37	19	9	5
9	8	J. SWARTZ	3	4	8	16	29	45	57	54	38	21	10	5
9	9	J. SWARTZ	4	7	12	20	33	45	53	50	39	21	9	5
9	10	J. SWARTZ	3	6	12	20	31	41	51	47	35	19	10	4
9	11	J. SWARTZ	4	6	12	18	30	39	50	47	36	19	9	4
9	12	J. SWARTZ	2	4	8	16	30	46	57	52	35	23	10	5
10	1	H. MEMAMI	9	12	15	20	31	42	49	49	40	28	19	12
10	2	H. MEMAMI	12	15	17	23	32	40	48	46	39	29	21	15
10	3	H. MEMAMI	12	14	17	22	32	40	44	41	34	28	19	15
10	4	H. MEMAMI	9	11	14	20	28	41	51	50	43	30	20	12
10	5	H. MEMAMI	13	14	16	22	31	38	39	33	31	22	16	13
10	6	H. MEMAMI	6	8	14	22	35	47	54	51	36	24	16	10
10	7	H. MEMAMI	9	10	14	20	30	38	39	39	35	23	16	12
10	8	H. MEMAMI	9	9	13	21	30	39	38	36	33	24	16	12
10	9	H. MEMAMI	10	11	15	21	30	37	37	36	36	24	16	12
10	10	H. MEMAMI	9	10	13	20	31	38	39	38	35	23	15	11
10	11	H. MEMAMI	10	10	13	22	31	36	37	36	32	23	14	10
10	12	H. MEMAMI	9	11	15	24	33	35	35	35	31	21	13	8

Table 4.5 Feature Vectors for Circular Autocorrelation Function Parameters

$$a_n = 1/2, M = 1, N = 12.$$

of parameter sets 2 and 4 are about equal--the limit here may be the dimension (MN) of the feature vector. It was decided not to investigate larger dimension feature vectors and on the basis of its recognition accuracy and simplicity, parameter set 2 was chosen to be used with the circular autocorrelation function in the remainder of this work.

4.3 Comparison of Some Classification Algorithms

In attempting to decide on the type of classifier to be used in this pattern recognition system, statistical classifiers were rejected because the feature vector sample size was too small to determine the necessary probability densities. Of the non-statistical classifiers the nearest neighbor rule and the distance-weighted k-nearest neighbor rule were selected as the most likely candidates to give good recognition accuracy. Starting with the distance-weighted k-nearest neighbor rule, a simple weight function was used:

$$w_{pq} = \frac{10^5}{(e_{pq} + 1)^T} \quad \text{for } 0 \leq e_{pq} \leq 500 \quad (4-6)$$

$$w_{pq} = 0 \quad \text{for } e_{pq} > 500$$

$$T = \sum_{*} \frac{10^3}{(e_{pq} + 1)} \quad \begin{array}{l} \text{*summed over the } k \\ \text{smallest } e_{pq} \text{'s } \leq 500. \end{array} \quad (4-7)$$

k was fixed at 10, so that only the ten lowest e_{pq} 's were used in the weight computation and therefore only the ten highest weights computed; all other weights being set to zero. This weight function will tend to give:

$$w_{pq} \rightarrow 100 \text{ for } e_{pq} \rightarrow 0$$

$$w_{pq} \rightarrow 0 \text{ for } e_{pq} \rightarrow 1000$$

$$w_{pq} \rightarrow w_{rs} \text{ for } e_{pq} \rightarrow e_{rs} .$$

The above relations show that this weight function satisfies the criteria for a distance-weighted k-nearest neighbor classifier weight function as expressed by Dudani [5]. The evidence of an authority file vector close to an unknown input vector should be weighted more heavily than the evidence of another authority file vector which is at a greater distance from the unknown. This is accomplished by having a weight function which varies with the distance between the unknown and authority file vector in such a manner that the weight decreases for increasing unknown to authority file vector distance. The above relations deteriorate as the distance between the unknown and the authority file vector increases (as e_{pq} approaches 1000). Any error (distance) above 500 is considered large enough to make any correspondence between the unknown and authority file vector unlikely and thus its corresponding weight is set to zero. With k set to 10, authority files of depth 3, and a feature extractor with parameter set $a_m = 1/2$, $M = 1$, and $N = 12$, the classifier was tested on facial profile images 1-6 of each class.

The authority file vectors were selected to give the best recognition accuracy on the same 60 images and the result is given

in Table 4.6 as test 1.

In examining the results of the experiment, it was noticed that the weight function seemed to assign too large values to terms with large errors. Correspondingly, the weight function was modified to;

$$w_{pq} = \frac{10^5}{((e_{pq})^2 + 1)T} \quad \text{for } 0 \leq (e_{pq})^2 \leq 500 \quad (4-8)$$

$$w_{pq} = 0 \quad \text{for } (e_{pq})^2 \geq 500$$

$$T = \sum_{*} \frac{10^3}{((e_{pq})^2 + 1)} \quad \begin{array}{l} \text{*summed over the } k \\ \text{smallest } e_{pq}^2 \text{'s } \leq 500 \end{array} \quad (4-9)$$

All other parameters were held constant and the experiment repeated. The result is given as test 2 in Table 4.6. A significant improvement in recognition accuracy was obtained, and therefore this weight function was used for all subsequent work.

With the weight function for the distance-weighted k-nearest neighbor classifier selected, a comparison of this classifier and the nearest neighbor classifier was performed. For the feature extractor parameters $a_m = 1/2$, $M = 1$ and $N = 12$, $k = 10$ for the distance-weighted k-nearest neighbor classifier, and an authority file depth of 3, both classifiers were trained and tested on the full set of 120 facial profile images. Training was accomplished using the activity sort algorithm discussed in Chapter III. The results of this experiment are shown in Tables 4.7 and 4.8. It can be seen that the results are very similar for the two classifiers, but influenced by the higher number of classes classified 100% correctly and the work of Dudani [5], the distance

Test Number	Weight Function	Recognition Accuracy	
		No. Correct	% Correct
1	Eq.(4-6)	45	75.0
2	Eq.(4-8)	49	81.7

Table 4.6 Weight Function Comparison for the k -Weighted Nearest Neighbor Rule.

Classifier Type	Recognition Accuracy		
	No. Correct	% Correct	No. of Classes 100% Correct
NN	98	81.7	4
k-NN	98	81.7	5

Table 4.7 Recognition Accuracy for Two Classifier Types

Classifier Type	Recognition Accuracy, % Correct/Class									
	1	2	3	4	5	6	7	8	9	10
NN	75	83	67	100	83	83	58	100	75	92
k-NN	58	83	67	83	100	83	67	100	75	100

Table 4.8 Recognition Accuracy per Class for Two Classifier Types

Feature Vector Component, n	Mean	Standard Deviation
1	5.01	2.72
2	6.65	2.43
3	12.03	2.76
4	21.87	3.76
5	35.98	5.75
6	47.80	8.48
7	51.74	10.34
8	44.48	8.89
9	32.91	6.15
10	19.93	3.21
11	11.67	3.27
12	7.38	2.96

Table 4.9 Feature Vector Component Means and Standard Deviations

weighted k-nearest neighbor classifier was chosen to be used in the remainder of this work.

In this discussion of classification algorithms, no mention has been made of feature vector normalization. Feature vector normalization was not used in this work for two reasons. First, from the means and standard deviations of the facial image vector components of Table 4.3, given in Table 4.9, it can be seen that the standard deviations are all of the same order of magnitude, indicating that a component-by-component normalization would yield little, if any, gain in recognition accuracy. Second, there is no evidence that a vector length normalization was warranted. A vector length normalization would leave the classification dependent only upon the angles between the unknown and the authority file vectors.

Based upon the results of the experiments described in this chapter, the circular autocorrelation parameters were set at $a_m = 1/2$, $M = 1$ and $N = 12$. It was noticed that for weight equations (4-8) and (4-9) usually only the first 5 nearest neighbors had significant weights in the k-weighted nearest neighbor classifier. The distance weighted k-nearest neighbor classifier parameters were thus selected as weight equations (4-8) and (4-9), and $k = 5$.

CHAPTER V

SUMMARY

5.1 Summary of Results

This thesis has presented a series of experiments on two-dimensional pattern recognition leading to a system capable of recognizing human facial profiles. A comparison of two feature extraction techniques was made, one of which has been used previously in an operating system, the other original to this work. Two deterministic classification algorithms, a piecewise linear classifier and a non-linear classifier, were compared. Two heuristic training rules for authority file vector selection were described and shown to provide decision surfaces for good class separability. The recognition accuracy of the final system was found to be at least as good as that of human recognizers presented with the same data.

The circular autocorrelation feature extraction technique developed in this thesis was shown to be invariant under image size change and translation, and to have predictable behavior under image rotation. Several experiments were performed to determine the optimum constants for this function and it was found that for a 12-dimensional feature vector a constant radius of one-half the square root of the image area and angle increments of 15 degrees gave optimum results on the test images. The radius constant was found to be not

critical, but for values less than one-half, the feature vectors seemed to have less variation from class to class; for values greater than one-half, several feature vector components were zero. In both cases the recognition accuracy was reduced.

The moment invariant feature extraction technique described by Hu [35] and used by Dudani [5] was also used for facial profile recognition. In all cases the maximum recognition accuracies obtained using a moment invariant feature extractor were significantly less than those obtained using a circular autocorrelation feature extractor. Considering the 95% recognition accuracy obtained by Dudani on aircraft identification using moment invariants, the 55-70% recognition accuracy obtained on facial profiles was disappointingly low. Three possible reasons for this low recognition accuracy may be advanced. The facial profile recognition problem was defined with 10 classes while the aircraft recognition problem had only 6. The increased number of classes increases the difficulty of class separation by the classifier and can thus reduce the recognition accuracy. Facial profiles are very similar with only subtle variations to determine one class from another, while in many views, different aircraft have distinctly different silhouettes. Facial profile recognition may therefore be a more difficult task. This assumption is supported by the recognition accuracy of humans at the two tasks. Human recognition accuracies were 73-76% correct for facial profiles and 79-92% correct for aircraft. The input system produced images with considerable variation from sample to sample within a given class. This means that test images

presented to the machine for recognition may vary considerably from the training image. This effect was less noticeable with the aircraft images because of the higher image/background contrast ratio, again indicating that the facial profile recognition problem is more difficult. In aircraft recognition, a pattern recognition system using the moment invariant feature extractor performed slightly better than humans (95% versus 79-92%), while for facial profile recognition the pattern recognition system using the moment invariant feature extractor performed slightly worse than humans (55-70% versus 73-76%).

Two weight functions for the distance-weighted k-nearest neighbor classifier were investigated. It was found that the weight function dependent upon the Euclidian distance squared (e_{pq}^2) gave about 7% better recognition accuracy than the weight function dependent upon Euclidian distance (e_{pq}). Using the weight function dependent upon e_{pq}^2 , it was found that only the first five nearest neighbors had significant weights assigned and therefore k was fixed at five instead of depending upon Euclidian distance. The nearest neighbor classification algorithm was compared against the distance-weighted k-nearest neighbor classifier. Although the significantly better recognition accuracy of the distance-weighted k-nearest neighbor classifier described by Dudani [5] was not observed, the distance-weighted k-nearest neighbor classifier did give a larger number of classes 100% correctly classified. There are two possible explanations for this. The weight function used in Dudani's classifier differs slightly from that used in this classifier, and Dudani used different training and test data, while in this comparison

the training data was also used as the test data.

Two heuristic training algorithms, the bubble sort rule and the activity sort rule were described and a rationale for their operation presented. The two sorting rules were compared in a simple experiment and the activity sort rule was found to give slightly better recognition accuracy. The number of samples necessary to train the authority files using the activity sort rule was found to be about three times the training set size. To verify that the activity sort rule did indeed select the authority file vectors to provide the best recognition accuracy on an independent test set, a large sample size experiment was performed. Using Dudani's aircraft feature vectors and pattern recognition system, the authority files were generated using the activity sort rule. After about two passes through the training set the recognition accuracy was tested with an independent test set. The recognition accuracy was found to be only slightly less than that obtained by Dudani (92% versus 95%), while the authority file size was about one-sixth that used by Dudani. These results were taken as verification of the expected operation of the activity sort rule for automatic authority file training.

The facial profile recognition system used the circular autocorrelation feature extractor, the distance-weighted k-nearest neighbor classifier and the activity sort rule for training. The 120 facial profile images were divided into training and test sets and the recognition accuracies for various compositions of training and test sets versus authority file depth were found. The results showed that the

maximum recognition accuracy on the test set did not occur at maximum file depth, but occurred around one-half the maximum file depth. The results also indicated that maximum recognition accuracy is achieved by using a training set large enough to contain the less probable feature vector variations on any class, resulting in a better defined decision surface. Recognition accuracies on the order of 80% (76-86%) were achieved in several instances and a maximum recognition accuracy of 90% was achieved using test images independent from the training images. When three human recognizers were presented with the same data, their maximum recognition accuracy was 73-76%.

5.2 Extensions of This Research

Like any research, this work has raised more questions than it has answered. Indeed, every aspect of this work requires further investigation. The circular autocorrelation function has been shown to work very well, even better than moment invariants, on this particular problem. Since circular autocorrelation is computationally both simple and fast, its performance on other problems might very well be worth investigating. Dudani [5] has shown that the distance-weighted k-nearest neighbor rule can perform significantly better than the nearest neighbor rule. This was not observed in this work, and an investigation into the reasons for this may be warranted, since the nearest neighbor rule is a simpler classification algorithm.

The two automatic training rules described in this thesis need a much more exhaustive treatment. No attempt was made here to

provide a theoretical explanation of their performance--this omission should be corrected. These two rules seem to be very useful for authority file vector selection, their applicability to other pattern recognition systems with other feature vector types should be investigated. The experiment using Dudani's data and the activity file sort rule described in Section 3.5 should be repeated with smaller authority file depths to determine how the activity sort rule performs under these conditions and just which feature vectors are used.

For facial recognition, several hardware improvements need to be made. The most glaring fault of the hardware used was the television camera's inability to reproduce a facial profile accurately. A more sophisticated input device, one able to distinguish between flesh tones and hair, clothing, and other background colors, is necessary. Probably the best way to do this is with a color television camera with the capability of selecting one range of colors and rejecting all others. Color information may also prove useful in the identification process.

Jagadeesa [45] has recently developed a color television camera interface for the Ohio State University Department of Electrical Engineering PDP-9 computer, and it is to be expected that results of the use of color information in this and other image recognition problems will be forthcoming. The prefiltering, edge extraction, and feature vector extraction could be speeded up considerably by implementing a binary array processor in hardware rather than simulation [41].

Besides hardware, other facial images should be investigated. With the proper input system, repeatable binary images of the front view

of the face should be possible. Such images may have the potential for providing more information than the profile. Even more information may be obtained by using several binary images with different thresholds to obtain a gray scale. The problem of automatic selection of a face from a scene has been totally ignored. This problem may have to be solved before a practical recognition system can be produced.

This work along with others [5,6,7] has shown that pattern recognition of two-dimensional images by machine is possible on a well defined problem in a laboratory environment. Recognition accuracies in excess of those achieved by humans have been demonstrated. The next step is to remove the machine from the laboratory and apply this knowledge to develop systems for less ideal conditions.

APPENDIX A: SELECTED FACIAL PROFILES

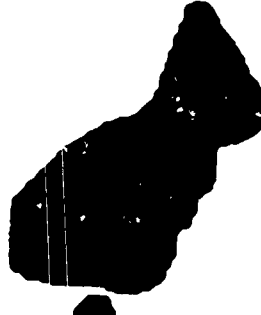
This appendix consists of twenty human facial profile images; two images per class. The images were photographed from the Tektronix 611 display after they had been filtered to remove system noise and produce a smoothed boundary. Except for class 8, the images were not selected to show the maximum variation from sample to sample within a given class; they are typical images and demonstrate the typical variations from sample to sample for this input system. Class 8 contains a reduced size facial profile that was used to check the recognition systems' size invariance.



Class 1



Class 2



Class 3

Reproduced from
best available copy





Class 4



Class 5



Class 6



Class 7

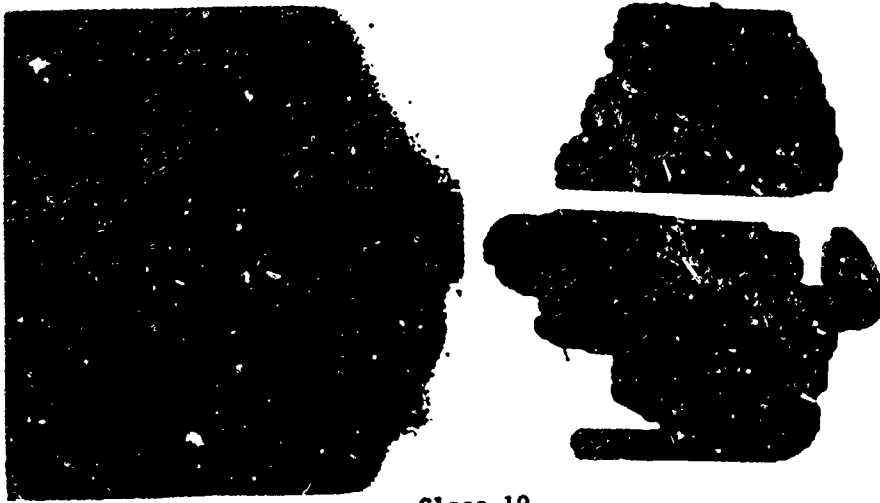


Class 8



Class 9





Class 10

APPENDIX B: SELECTED PROFILES AFTER EDGE EXTRACTION

This appendix consists of the same twenty facial profiles of Appendix A, but after the front edge of the profile has been extracted. It can be seen that the sample-to-sample variation within a given class is less than that of the full profile, mainly because of the removal of the hairline. Width variations in the extracted edge due to area changes in the full profile are not noticeable (see Section 2.3).



Class 1



Class 2



Class 3



Class 4



Class 5



Class 6



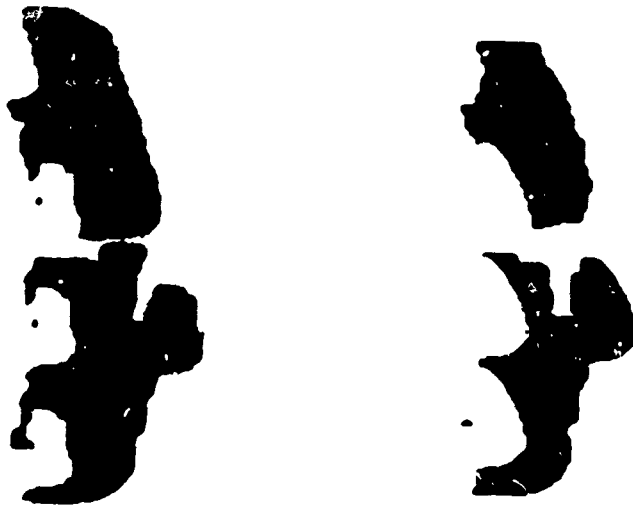
Class 7



Class 8



Class 9



Class 10

APPENDIX C: COMPUTER PROGRAMS

This appendix contains the more pertinent programs written for this thesis. The programs are written in FORTRAN IV, except for three bit manipulation programs and a random number generator that are written in PDP-9 assembler. There are three main routines in this appendix. RECOG3 is an on-line facial profile recognition routine. Input is from either the television or a disk file. An image may be filtered, stored in a disk file, or both. An identification of the input image may be requested and various operations on the authority file are permitted. RECOG4 is the routine that was used to generate a file of feature vectors corresponding to the 120 facial profile images. The images had been filtered using RECOG3 before they were used with RECOG4. RECOG5 is the routine used to generate an authority file using either the bubble or activity sort rule. The recognition accuracy of the authority file may also be tested with this routine.

Because of the experimental nature of this work, the programs were written as short subroutines that could be called by the various main routines to avoid duplicating large quantities of code. Each subroutine was designed to perform one well defined task. CRCT1 corrects a hardware flaw in the television interface by adding some bits that are not input and deleting some noise points on the border of the image. FILOP2 allows manipulation of a vector file. The file may be listed or cleared, or a vector may be entered or deleted.

FLTR2 removes small noise clusters and smooths the boundary of a binary image. IDENT2 performs a distance weighted k-nearest neighbor classification of an unknown feature vector for some specified authority file. RANDOM is a random number generator. SORT1 is an implementation of the bubble sort training rule. SORT2 is an implementation of the activity sort training rule. XFRM3 performs the circular autocorrelation function on a binary image, generating the 12-dimensional feature vector. XTRCT1 extracts the right-hand (zero degree) edge of an image, i.e., the front edge of a facial profile.

Three array processing routines were written in assembler to complement those written by Miller [3]. INTRSC counts the number of corresponding one-cells in two binary arrays. INVERT complements every cell of a binary array. SHIFT translates a binary image a specified distance along the x and y axes.

```

C ARRAY PROCESSING ROUTINE RECOG3
C G. KAUFMAN 1/74
C THIS ROUTINE TAKES A 360 X 240 BINARY IMAGE FROM DISK6 OR THE
C TV CAMERA AND IDENTIFIES THE IMAGE FROM AN IMAGE FILE.
C
    REAL C,CMND,D,F,NAME(3),NAMES(10,3),R,S,T
    INTEGER AREA,CLS,DFL(1740),DFLN,ERR,FN,FTH(12),FTRFL(10,12,12),
    1 IDN,IMGN,IMG1(20,240),I1,I2,NL,SMPL,SW1,TYP(10,12,2)
    LOGICAL IEXIST
    EQUIVALENCE (DFL(1),NAMES),(DFL(61),TYP),(DFL(301),FTRFL)
    DATA C/1HC/,D/1HD/,F/1HF/,R/1HR/,S/1HS/,T/1HT/
C SELECT AUTHORITY FILE DEPTH
    SW1=4
C PRINT HEADING, GET DFL
    WRITE(4,12)
    FORMAT (32H ARRAY PROCESSING ROUTINE RECOG3 /
    1 20H SSW? - OUTPUT ARRAY / 20H SSW1 - OUTPUT ON LP /
    2 21H SSW3 - IDENT2 OUTPUT/10H COMMANDS:9H CONTINUE/
    3 11H DISK INPUT/18H FILE MANIPULATION/8H RESTART/
    4 11H SAVE IMAGE/9H TV INPUT)
    290 WRITE(4,82)
    300 FORMAT (21H ENTER FILE NO. IN 12)
    READ(4,90) DFLN
    CALL DKUNIT (5)
    CALL DKLOAD (DFL,DFLN)
C SELECT ACTION
    600 WRITE(4,27)
    200 FORMAT (16H+READY FOR INPUT)
    200 WRITE(4,100)
    100 FORMAT (2H+>)
    READ(4,30) CMND
    300 FORMAT (A1)
    IF (CMND.EQ.C) GO TO 70
    IF (CMND.EQ.D) GO TO 40
    IF (CMND.EQ.F) GO TO 280

```



```

140 IF (CMND.EQ.H) GO TO 290
    IF (CMND.EQ.S) GO TO 420
    IF (CMND.EQ.T) GO TO 400
    WRITE(4,140)
    FORMAT (14H+INVALID INPUT)
    GO TO 202
C FEATURE FILE OPERATIONS
280 CALL FILOP2 (NAMES,TYP,FTRFL,SMPL,FTR,DFLN,SW1)
    CALL OKSTOR (DFL,DFLN)
    GO TO 202
C INPUT FROM TV
400 CALL TVLOAD (IMG1,NL)
    CALL OKUNIT (5)
    CALL CRCT1 (IMG1)
    IMGN=0
    GO TO 112
C INPUT FROM DISK
40 WRITE(4,82)
    READ(4,93) FN
    FORMAT (12)
    IF ((FN.LE.0).OR.(FN.GE.64)) GO TO 150
    CALL OKUNIT (6)
    IF (IEXIST(FN)) GO TO 410
    WRITE(4,140)
    GO TO 202
410 IMGN=FN
    CALL OKLOAD (IMG1,FN)
C DISPLAY INPUT
110 CALL ERASE
    CALL DISPLAY (IMG1)
    WRITE(4,57)
    FORMAT (16H+INPUT COMPLETED)
    GO TO 202
50
C STORE IMAGE ON DISK
420 WRITE(4,82)
    READ(4,93) FN

```

```

IF ((FN.GT.0).AND.(FN.LT.64)) GO TO 130
WRITE(4,140)
GO TO 200
130  CALL OKUNIT (6)
      CALL OKSTOR (IMG1,FN)
      WRITE(4,220)
220  FORMAT (17H+OUTPUT COMPLETED)
      GO TO 200
C  FILTER OR IDENTIFY
70  IF (IMG1(20,240).EQ.1) GO TO 240
      CALL FLTR2 (IMG1)
      IMG1(20,240)=1
190  CALL ERASE
      CALL DISPLAY (IMG1)
      WRITE(4,260)
260  FORMAT (20H+FILTERING COMPLETED)
      GO TO 200
C  EXTRACT FRONTAL FEATURES
240  CALL INTRSC (IMG1,IMG1,AREA)
      CALL XTCT1 (IMG1,AREA,0)
C  GET FEATURE VECTOR
      CALL XFRM3 (IMG1,AREA,FTR)
C  OUTPUT RESULTS OF CORRELATION ON TTY AND PDS
      CALL ERASE
      CALL DISPLAY (IMG1)
      WRITE(1,120) IMGN,AREA,FTR
      IF (ISENSA(1).EQ.0) GO TO 180
      WRITE(6,120) IMGN,AREA,FTR
120  FORMAT (10H INPUT NO. 2X,5H AREA 10X,15H FEATURE VECTOR /
116,111,1X,615 / 18X,615 /)
C  IDENTIFICATION BEGINNING

```

```

180      CALL IDENT2 (FTR,FTRFL,TYP,CLS,SMPL,SW1)
      WRITE(1,160) (NAMES(CLS,I1),I1=1,3),SMPL
      IF (ISENSW(1).EQ.0) GO TO 60
      WRITE(6,160) (NAMES(CLS,I1),I1=1,3),SMPL
      FORMAT (25H MOST LIKELY SUSPECT IS: 3A5,2X,9H SAMPLE= I3/)
      GO TO 60
      END
160

```

```

C  ARRAY PROCESSING ROUTINE RECOG4
C  G. KAUFMAN 2/74
C  THIS ROUTINE OBTAINS THE FEATURE VECTORS OF SIX IMAGES ON DISK 6
C  AND STORES THE RESULTS IN FILE02 OMP ON DISK 5.
C

```

```

      REAL NAME(3),NAMES(10,3)
      INTEGER AREA,DFL(1740),FN,FTR(12),FTRFL(10,12,12),
      1 I1,I2,IMG1(20,240),IMGN,SMP,SMPL,TYP(10,12,2)
      EQUIVALENCE (DFL(1),NAMES),(DFL(61),TYP),(DFL(301),FTRFL)
C  LOAD FEATURE FILE

```

```

      CALL DKUNIT (5)
      CALL DKLOAD (DFL,2)
C  MODIFY FILE ?
80      CALL FILOP2 (NAMES,TYP,FTRFL,SMPL,FTR,2,12)
C  GET FILE, IMAGE NUMBERS, IMAGE NAME

```

```

      WRITE(4,10)
      FORMAT (29H ENTER FEATURE FILE NO. IN I2)
10      READ(4,20) FN
      FORMAT (I2)
20      WRITE(4,90)
      FORMAT (23H ENTER SAMPLE NO. IN I2)
90      READ(4,22) SMPL

```

```

30  WRITE(4,30)
    FORMAT (17H ENTER IMAGE NAME)
40  READ(4,40) NAME
    FORMAT (3A5)
50  WRITE(4,50)
    FORMAT (27H ENTER IMAGE FILE NO. IN I2)
    READ(4,20) IMGN
    DO 60 I1=1,3
60  NAMES(FN,I1)=NAME(I1)
    CALL DKUNIT (6)
    DO 70 I1=1,6
    SMP=I1+SMP-1
    TYP(FN,SMP,1)=SMP
    C LOAD FILE
      I2=IMGN+I1-1
      CALL DKLOAD (IMG1,I2)
      CALL ERASE
      CALL DISPLAY (IMG1)
    C EXTRACT PROFILE
      CALL INTRSC (IMG1,IMG1,AREA)
      CALL XTCT1 (IMG1,AREA,0)
      CALL ERASE
      CALL DISPLAY (IMG1)
    C TRANSFORM AND STORE
      CALL XFRM3 (IMG1,AREA,FTR)
70  DO 70 I2=1,12
      FTRFL(FN,SMP,I2)=FTR(I2)
      CALL DKUNIT (5)
      CALL DKSTOR (DFL,2)
      GO TO 80
    END

```

```

C TRAINING ROUTINE RECOG5
C G. KAUFMAN 2/74
C THIS ROUTINE TRAINS THE IDENTIFICATION ROUTINE IDENT2
C FROM STORED FEATURE VECTORS. INPUT IS FROM FILE#1 DMP ON DISK 6.
C
  REAL A,C,F,CMND,NAME(3),NAMES1(10,3),NAMES2(10,3),R,S,T
  INTEGER ACTV,AREA,CLS,DFLN,DFL1(1740),DFL2(1740),ERR,FTR(12),
  1 FTRFL1(10,12,12),FTRFL2(10,12,12),IDX1,IDX2,11,12,13,14,15,
  2 LIM1,LIM2,SMPL,SW1,SW2,SW3,TYP:(10,12,2),TYP2(10,12,2)
  EQUIVALENCE (DFL1(1),NAMES1),(DFL1(61),TYP1),(DFL1(301),FTRFL1)
  EQUIVALENCE (DFL2(1),NAMES2),(DFL2(61),TYP2),(DFL2(301),FTRFL2)
  DATA A/1HA/,C/1HC/,F/1HF/,R/1HR/,S/1HS/,1/1HT/

C SELECT SAMPLE NUMBER
  SW2=6

C OUTPUT HEADING
  LIM1=1
  LIM2=6
  WRITE(4,300)
300  FORMAT (24H TRAINING ROUTINE RECOG5/27H SSH1 - IDENT2 OUTPUT ON LP
      1 /27H SSW2 - RECOG5 OUTPUT ON LP/21H SSW3 - OUTPUT ON PDS/
      2 10H COMMANDS:/9H ACCURACY/9H CONTINUE/13H FEATURE FILE/
      3 6H STORE/6H TRAIN/)

C FIND PARAMETERS
340  WRITE(4,310)
310  FORMAT (27H+ENTER FEATURE FILE NO. IN 12)
200  READ(4,200) DFLN
      FORMAT (12)
      WRITE(4,400)
400  FORMAT (32H+ENTER AUTHORITY FILE DEPTH IN 12)
      READ(4,200) SW3

C INPUT FILES
      CALL DKUNIT (5)
      CALL DKLOAD (DFL1,1)
      CALL DKLOAD (DFL2,DFLN)

```

```

C SELECT ACTION
180 WRITE(4,130)
130 FORMAT (2H+>)
    READ(4,140) CMND
140 FORMAT (A1)
    IF (CMND.EQ.A) GO TO 150
    IF (CMND.EQ.C) GO TO 230
    IF (CMND.EQ.F) GO TO 320
    IF (CMND.EQ.R) GO TO 340
    IF (CMND.EQ.S) GO TO 160
    IF (CMND.EQ.T) GO TO 150
    WRITE(4,170)
170 FORMAT (14H+INVALID INPUT)
    GO TO 180
C FEATURE FILE OPERATIONS
320 CALL FILOP2 (NAMES2,TYP2,FTRFL2,SMPL,FTR,DFLN,SW3)
    GO TO 180
C STORE AUTHORITY FILE
160 CALL DKSTOR (DFL2,DFLN)
    GO TO 340
C ACCURACY TEST
150 IF (ISENSW(2).EQ.0) GO TO 60
    WRITE(6,100)
100 FORMAT (24H1TRAINING ROUTINE RECOG5/)
C SELECT LIMITS
60 IF (CMND.EQ.T) GO TO 230
    WRITE(4,380)
380 FORMAT (25H+ENTER START SAMPLE IN I2)
    READ(4,200) LIM1
    WRITE(4,390)
390 FORMAT (23H+ENTER END SAMPLE IN I2)
    READ(4,200) LIM2
230 DO 10 I1=1,10
    DO 10 I2=LIM1,LIM2
    IF (CMND.EQ.A) GO TO 220
    IF (CMND.NE.T) GO TO 330

```

```

C RANDOM VECTOR SELECTION
15=0
350 15=15+1
    CALL RANDOM (RND)
    I1=IABS(MOD(RND,10))+1
    CALL RANDOM (RND)
    I2=IABS(MOD(RND,SW2))+1
    GO TO 22

C SELECT VECTOR
330 WRITE(4,190)
190 FORMAT (17H+TYPE CLASS IN I2)
    READ(4,200) I1
    WRITE(4,210)
210 FORMAT (16H+TYPE SMPL IN I2)
    READ(4,200) I2

C GET VECTOR
220 DO 20 I3=1,12
20  FTR(I3)=FTRFL1(I1,I2,I3)
C ATTEMPT IDENTIFICATION
    CALL IDENT2 (FTR,FTRFL2,TYP2,CLS,SMPL,SW3)
    IF (CLS.EQ.I1) GO TO 70
C INCORRECT IDENTIFICATION, ENTER VECTOR
    IF (ISENSW(3).EQ.0) GO TO 280
    WRITE(1,80) I1,TYP1(I1,I2,1),CLS,SMPL
    IF (ISENSW(2).EQ.0) GO TO 280
    WRITE(6,90) I1,TYP1(I1,I2,1),CLS,SMPL
80  FORMAT (/30H INCORRECT IDENTIFICATION. I1=,I3,5X,
1 4H I2=,I3,5X,7H CLASS=,I3,5X,6H SMPL=,I3)
280 DO 50 I3=1,3
50  NAME(I3)=NAMES1(I1,I3)
    IF (CMND.EQ.A) GO TO 370
    CALL SORT2 (NAMES2,TYP2,FTRFL2,NAME,I1,TYP1(I1,I2,1),FTR,1,SW3)
C OUTPUT MODIFIED FILE
370 IF (ISENSW(3).EQ.0) GO TO 120
    WRITE(1,40)

```

```

40      IF (ISENSW(2).EQ.0) GO TO 120
        WRITE(6,40)
        FORMAT ( 6H CLASS,6H SMPL,3X,
15H NAME,13X,9H ACTIVITY,27X,15H FEATURE VECTOR / 46X,1H1,
25X,1H2,5X,1H3,5X,1H4,5X,1H5,5X,1H6,5X,1H7,5X,1H8,5X,
31H9,4X,2H10,4X,2H11,4X,2H12 /)
120     DO 360 I4=1,SW3
        IF (ISENSW(3).EQ.0) GO TO 360
        WRITE(1,110) I1,TYP2(I1,I4,1),(NAMES2(I1,I3),I3=1,3),
17YP2(I1,I4,2),(FTRFL2(I1,I4,I3),I3=1,12)
        IF (ISENSW(2).EQ.0) GO TO 360
        WRITE(6,110) I1,TYP2(I1,I4,1),(NAMES2(I1,I3),I3=1,3),
17YP2(I1,I4,2),(FTRFL2(I1,I4,I3),I3=1,12)
        FORMAT (1H ,I4,I6,5X,3A5,2I8,11I6)
        CONTINUE
110     GO TO 240
360
C CORRECT IDENTIFICATION
70      IF (ISENSW(3).EQ.0) GO TO 30
        WRITE(1,90) I1,TYP1(I1,I2,1),SMPL
        IF (ISENSW(2).EQ.0) GO TO 240
        WRITE(6,90) I1,TYP1(I1,I2,1),SMPL
90      FORMAT (/28H CORRECT IDENTIFICATION, I1,I3,5X,
14H I2=,I3,5X,6H SMPL=,I3 )
30      CALL SORT2 (NAMES2,TYP2,FTRFL2,NAME,CLS,SMPL,FTR,0,SW3)
240     IF (CMND.EQ.A) GO TO 10
        IF (CMND.NE.T) GO TO 180
        IF (I5.LE.250) GO TO 350
        GO TO 180
10      CONTINUE
        GO TO 180
        END

```



```

C  ARRAY PROCESSING ROUTINE CRCT1
C  G. KAUFMAN 1/74
C  THIS ROUTINE ADDS THE BITS NOT INPUT BY THE TV INTERFACE.
C  AND DELETES THE BORDER NOISE FOR 360 X 240 BINARY ARRAYS.
C
      SUBROUTINE CRCT1 (AR1)
      REAL C,CMND,I
      INTEGER AR1(20,240),AR2(20,240),I1,I2,NL
      COMMON /SCRATCH/AR2
      DATA C/1HC/,I/1HI/
C  SELECT ACTION
      WRITE(4,10)
      FORMAT (25H+COMMANDS: CONTINUE,INPUT)
10  READ(4,20) CMND
20  FORMAT (A1)
      IF (CMND.EQ.C) GO TO 30
      IF (CMND.EQ.I) GO TO 50
      WRITE(4,60)
60  FORMAT (14H+INVALID INPUT)
      GO TO 70
C  GET MISSING BITS
50  CALL TVLOAD (AR2,NL)
      CALL INVERT (AR2)
      CALL DKSTOR (AR2,57)
C  INSERT MISSING BITS
30  CALL DKLOAD (AR2,57)
      CALL CHANGE (AR1,1,AR2,1,AR2,1)
C  DELETE BORDER NOISE
      DO 40 I1=1,20
      DO 40 I2=1,240
      IF ((I2.LT.20).OR.(I2.GT.220)) AR1(I1,I2)=0
      IF ((I1.LT.2).OR.(I1.GT.16)) AR1(I1,I2)=0
40  CONTINUE
      RETURN
      END

```

```

C FILE MANIPULATION ROUTINE FILOP2
C G. KAUFMAN 2/74
C THIS ROUTINE ALLOWS MODIFICATIONS OF THE FEATURE FILE.
C
SUBROUTINE FILOP2 (NAMES,TYP,FTRFL,SMPL,FTR,DFLN,SW1)
REAL BL,C,CMND,D,E,L,N,NAME(3),NAMES(10,3)
INTEGER DFLN,FTRFL(10,12,12),FTR(12),IDX1,IDX2,
1 I1,I2,I3,SW1,SMPL,TYP(10,12,2)
DATA BL/1H /,C/1HC/,D/1HD/,E/1HE/,L/1HL/,N/1HN/
C SELECT FILE OPERATION
WRITE(4,250)
250 FORMAT (41H+COMMANDS: CONTINUE,DELETE,ENTER,LIST,NEW)
280 WRITE(4,40)
40 FORMAT (2H++)
READ(4,30) CMND
30 FORMAT (A1)
IF (CMND.EQ.C) GO TO 60
IF (CMND.EQ.D) GO TO 310
IF (CMND.EQ.E) GO TO 390
IF (CMND.EQ.L) GO TO 440
IF (CMND.EQ.N) GO TO 290
WRITE(4,140)
140 FORMAT (14H+INVALID INPUT)
GO TO 280
C LIST FEATURE FILE
440 WRITE(6,260) DFLN
260 FORMAT (1H1,5H FILE,I2,11H DMP DISK 5 / 6H CLASS,1X,5H SMPL,
1 3X,5H NAME,13X,9H ACTIVITY,27X,15H FEATURE VECTOR /
2 46X,1H1,5X,142,5X,1H3,5X,1H4,5X,1H5,5X,1H6,5X,1H7,5X,1H8,
3 5X,1H9,4X,2H10,4X,2H11,4X,2H12/)
DO 80 I1=1,10
DO 80 I2=1,12
80 WRITE(6,270) I1,TYP(I1,I2,1),(NAMES(I1,I3),I3=1,3),TYP(I1,I2,2),
1 (FTRFL(I1,I2,I3),I3=1,12)

```

```

270  FORMAT (1H ,14,16,5X,3A5,2I8,11I6)
      GO TO 280
C  CLEAR FEATURE FILE
290  DO 300 I3=1,12
      DO 300 I2=1,12
      TYP(I3,I2,1)=0
      TYP(I3,I2,2)=0
      DO 300 I1=1,12
      FTRFL(I3,I2,I1)=0
      DO 300 I1=1,12
      NAMES(I3,I1)=BL
      GO TO 280
C  DELETE VECTOR FROM FEATURE FILE
310  WRITE(4,17)
10  FORMAT (17H+TYPE CLASS IN I2)
      READ(4,20) I2
      WRITE(4,50)
      FORMAT (16H+TYPE SMPL IN I2)
      READ(4,20) I3
      DO 90 I1=1,12
      IF (TYP(I2,I1,1).NE.I3) GO TO 90
      I3=I1
      GO TO 100
90  CONTINUE
100  TYP(I2,I3,1)=0
      TYP(I2,I3,2)=0
      DO 370 I1=1,12
      FTRFL(I2,I3,I1)=0
      GO TO 280
C  ENTER VECTOR IN FEATURE FILE
390  WRITE(4,10)
      READ(4,20) I1
      FORMAT (I2)
      WRITE(4,170)

```

```

170  FORMAT (23H+TYPE IMAGE NAME IN A15)
    READ(4,182) NAME
180  FORMAT (3A5)
    CALL SORT2 (NAMES,TYP,FTRFL,NAME,I1,SMPL,FTR,1,SW1)
    GO TO 282
    RETURN
    END

C  ARRAY PROCESSING ROUTINE FLTR2
C  G. KAUFMAN 1/74
C  THIS ROUTINE ELIMINATES BIT CLUSTERS WITH RADII LESS THAN
C  1.5 CELLS FROM THE INPUT 360 X 240 BINARY ARRAY.
C
C  SUBROUTINE FLTR2 (AR1)
C  INTEGER AR1(20,240),AR2(20,240),NC
C  COMMON /SCRATCH/AR2
C  REMOVE CELLS NOT COMPLETELY SURROUNDED
C  CALL RESET (AR2)
C  CALL THRESH (AR1,2,8,511,0,AR2,1,NC)
C  IF (ISENSW(0).EQ.0) GO TO 10
C  CALL ERASE
C  CALL DISPLAY (AR2)
C  EXPAND IMAGE TWICE
10  CALL RESET (AR1)
    CALL THRESH (AR2,1,1,186,0,AR1,1,NC)
    CALL THRESH (AR1,1,1,511,0,AR2,1,NC)
    CALL CHANGE (AR1,1,AR2,1,AR2,1)
    RETURN
    END

```

```

C FILE SEARCH ROUTINE IDENT2
C G. KAUFMAN 2/74
C THIS ROUTINE SEARCHES FILE FTRFL TO OBTAIN THE SW1 BEST MATCHES
C WITH VECTOR FTR. FTR WRAPS AROUND - ONLY 0 DEG. VECTORS ARE
C SEARCHED.
C
C SUBROUTINE IDENT2 (FTR,FTRFL,TYP,CLS,SMPL,SW1)
C INTEGER CLS,ERR1,ERRS(10),FTR(12),FTRFL(10,12,12),
C 1 IDNS(2,10),IDX1,IDX2,I1,12,I3,I4,WGHT1,WGHTS(2,10),
C 2 SMPL,SW1,SW2,TOT,TYP(10,12,2)
C SELECT NEIGHBORHOOD SIZE
C SW2=5
C RESET WGHTS,ERRS
C DO 260 I1=1,SW2
C WGHTS(1,I1)=0
C ERRS(I1)=100000
260 C FILE FEATURE VECTOR LOOP
C DO 30 I1=1,10
C DO 30 I2=1,SW1
C COMPUTE DISTANCE SQUARED
C ERR1=0
C DO 20 I3=1,12
C ERR1=ERR1+(FTRFL(I1,I2,I3)-FTR(I3))**2
20 CONTINUE
C IF DISTANCE LESS THAN PREVIOUS VALUES RECORD PRESENT CLASS,SMPL,ERR
C DO 60 I3=1,SW2
C IF (ERR1.GE.ERRS(I3)) GO TO 60
C PUSH DOWN STACKS
C I4=0
C I4=I4+1
C IF ((SW2+1-I4).EQ.I3) GO TO 80
C IDX1=SW2+1-I4
C IDX2=SW2-I4
C ERRS(IDX1)=ERRS(IDX2)
C IDNS(1,IDX1)=IDNS(1,IDX2)
C IDNS(2,IDX1)=IDNS(2,IDX2)
70

```

```

      GO TO 70
C   INSERT NEW CLASS, ERROR, SAMPLE
80   ERRS(I3)=ERR1
      IDNS(1,I3)=I1
      IDNS(2,I3)=TYP(I1,I2,1)
      GO TO 30
60   CONTINUE
30   CONTINUE
C   FIND WEIGHTS
220  TOT=0
      DO 90 I1=1,SW2
      IF (ERRS(I1).GE.500) GO TO 90
      TOT=TOT+1000/(ERRS(I1)+1)
90   CONTINUE
      DO 100 I1=1,SW2
      IF (ERRS(I1).GE.500) GO TO 100
      WGHTS(1,I1)=1000/(ERRS(I1)+1)*100/TOT
100  CONTINUE
C   FIND MOST LIKELY SUSPECT
C   SUM WEIGHTS OF SAME CLASS
      DO 110 I1=1,SW2
      WGHTS(2,I1)=WGHTS(1,I1)
      DO 110 I2=1,SW2
      IF (I2.EQ.I1) GO TO 110
      IF (IDNS(1,I1).NE.IDNS(1,I2)) GO TO 110
      WGHTS(2,I1)=WGHTS(2,I1)+WGHTS(1,I2)
110  CONTINUE
C   SELECT LARGEST WEIGHT
      WGT1=0
      DO 120 I1=1,SW2
      IF (ERRS(I1).GE.500) GO TO 120
      IF (WGHTS(2,I1).LE.WGT1) GO TO 120
      I2=I1
      WGT1=WGHTS(2,I1)
120  CONTINUE

```

```

C  OUTPUT SUSPECTS AND WEIGHTS
  IF (ISENSK(3).EQ.0) GO TO 10
  CALL ERASE
  CALL LINE11 (0,1,1000,0,0,1,0)
  WRITE(1,130)
  IF (ISENSK(1).EQ.0) GO TO 300
  WRITE(6,130)
  FORMAT (6H0CLASS,5X,5H SMPL,4X,6H ERROR,4X,7H WEIGHT)
  WRITE(1,140) ((IDNS(I3,I1),I3=1,2),ERRS(I1),
  1 WGHTS(1,I1),I1=1,SW2)
  IF (ISENSK(1).EQ.0) GO TO 10
  WRITE(6,140) ((IDNS(I3,I1),I3=1,2),ERRS(I1),
  1 WGHTS(1,I1),I1=1,SW2)
  FORMAT (15,3I10)
C  SET UP OUTPUT PARAMETERS
  10  CLS=IDNS(1,I2)
     SMPL=IDNS(2,I2)
     RETURN
     END

```

RANDOM	.TITLE RANDOM	
	.GLOBLE RANDOM,.DA	
	0	
	JMS*	.DA
	JMP	.+2
N	.OSA	4
	LAC	FLAG
	SZA	
	JMP	S1
	LAC	(1
	DAC	FLAG
	LAC	(200571
	DAC*	N
S1	LAC*	N
	AND	(4001
	RCR	
	SZL	
	JIP	S2
	SNA	
	JMP	S3
S4	LAC*	N
	RCR	
	YAD	(400000
	DAC*	N
	JMP*	RANDOM
S3	LAC*	N
	RCR	
	DAC*	
	JMP*	RANDOM
S2	SNA	
	JMP	S4
	JMP	S3
FLAG	0	
	.END	


```

C FILE MANIPULATION ROUTINE SORT1
C G. KAUFMAN 2/74
C THIS ROUTINE SORTS THE FEATURE FILE USING A BUBBLE SORT
C AND ENTERS NEW VECTORS.
C
SUBROUTINE SORT1 (NAMES, TYP, FTRFL, NAME, CLS, SMPL, FTR, SW1, DUM)
REAL NAME(3), NAMES(10,3)
INTEGER ACTV, CLS, DUM, FTR(12), FTRB(12), FTRFL(10,12,12),
1 I1, I2, I3, I0X1, I0X2, SMPL, SW1, TYP(10,12,2)
IF (SW1.EQ.1) GO TO 10
C BUBBLE SELECTED SAMPLE
C FIND SAMPLE
DO 20 I1=1,12
IF (TYP(CLS,I1,1).NE.SMPL) GO TO 20
I2=I1
GO TO 13
CONTINUE
C SAVE ENTRIES
130 DO 70 I1=1,12
70 FTRB(I1)=FTRFL(CLS,I2,I1)
ACTV=TYP(CLS,I2,2)
C PUSH DOWN STACKS
DO 30 I1=1,12
I0X1=I2+1-I1
I0X2=I2-I1
IF (I0X2.EQ.0) GO TO 40
DO 50 I3=1,12
FTRFL(CLS,I0X1,I3)=FTRFL(CLS,I0X2,I3)
TYP(CLS,I0X1,1)=TYP(CLS,I0X2,1)
TYP(CLS,I0X1,2)=TYP(CLS,I0X2,2)
C RENTER VECTOT, SMPL, ACTV
40 DO 60 I3=1,12
60 FTRFL(CLS,1,I3)=FTRB(I3)
TYP(CLS,1,1)=SMPL
TYP(CLS,1,2)=ACTV
GO TO 80

```

```

C  INSERT VECTOR
C  PUSH DOWN STACKS
10  DO 100 I1=1,11
    IDX1=13-I1
    IDX2=12-I1
    DO 90 I2=1,12
        FTRFL(CLS,IDX1,I2)=FTRFL(CLS,IDX2,I2)
        TYP(CLS,IDX1,I1)=TYP(CLS,IDX2,I1)
100  TYP(CLS,IDX1,2)=TYP(CLS,IDX2,2)
C  ENTER VECTOR, NAME, SMPL, ACTV
    DO 110 I1=1,12
        FTRFL(CLS,I1,I1)=FTR(I1)
    DO 120 I1=1,3
        NAMES(CLS,I1)=NAME(I1)
        TYP(CLS,I1,1)=SMPL
        TYP(CLS,I1,2)=0
    RETURN
    END
80

C  FILE MANIPULATION ROUTINE SORT2
C  G. KAUFMAN 2/74
C  THIS ROUTINE SORTS THE FEATURE FILE USING AN ACTIVITY SORT
C  AND ENTERS NEW VECTORS.
C
    SUBROUTINE SORT2 (NAMES,TYP,FTRFL,NAME,CLS,SMPL,FTR,SW1,SW2)
    REAL NAME(3),NAMES(10,3)
    INTEGER ACTV,CLS,FTR(12),FTRFL(10,12,12),I1,I2,
    1 SMPL,SW1,SW2,TYP(10,12,2)
    IF (SW1.EQ.1) GO TO 10
C  BUMP ACTIVITY
C  FIND SAMPLE
    DO 20 I1=1,12
        IF (TYP(CLS,I1,1).NE.SMPL) GO TO 20

```

```

20      I2=I1
      GO TO 80
      CONTINUE
80      TYP(CLS,I2,2)=TYP(CLS,I2,2)+1
      C  NORMALIZE ACTIVITIES
      IF (TYP(CLS,I2,2).LE.1000) GO TO 30
      DO 40 I1=1,12
      IF (TYP(CLS,I1,2).EQ.1) GO TO 40
      TYP(CLS,I1,2)=TYP(CLS,I1,2)/2
40      CONTINUE
      GO TO 30
      C  ENTER VECTOR
      C  FIND LOW ACTIVITY
10      ACTV=100000
      DO 50 I1=1,SW2
      IF (TYP(CLS,I1,2).GE.ACTV) GO TO 50
      ACTV=TYP(CLS,I1,2)
      I2=I1
      CONTINUE
50      C  ENTER VECTOR, NAME, SMPL, ACTV
      DO 60 I1=1,12
      FTRFL(CLS,I2,I1)=FTR(I1)
      DO 70 I1=1,3
      NAMES(CLS,I1)=NAME(I1)
      TYP(CLS,I2,1)=SMPL
      TYP(CLS,I2,2)=1
30      RETURN
      END

```

```

C ARRAY PROCESSING ROUTINE XFRM3
C G. KAUFMAN 1/74
C THIS ROUTINE USES A CIRCULAR AUTOCORRELATION FUNCTION
C TO OBTAIN A 12-D FEATURE VECTOR FROM THE 360 X 240
C INPUT IMAGE.
C
SUBROUTINE XFRM3 (AR1,AREA,FTR)
REAL COSINE(12),RAD,RAREA,SINE(12)
INTEGER AREA,AR1(20,240),AR2(20,240),FTR(12),I,NI,
1 XDISP,YDISP
COMMON /SCRATCH/AR2
DATA COSINE(1),COSINE(2),COSINE(3),COSINE(4),COSINE(5),
1 COSINE(6),COSINE(7),COSINE(8),COSINE(9),COSINE(10),
2 COSINE(11),COSINE(12)/1.,.96593,.86603,.70711,.5,.25882,
3 0.,-.25882,-.5,-.70711,-.86603,-.96593/
DATA SINE(1),SINE(2),SINE(3),SINE(4),SINE(5),SINE(6),
1 SINE(7),SINE(8),SINE(9),SINE(10),SINE(11),SINE(12)
2 /0.,.25882,.5,.70711,.86603,.96593,1.,.96593,
3 .86603,.70711,.5,.25882/
C FIND AREA OF IMAGE AND COMPUTE AUTOCORRELATION RADIUS
CALL INTRSC (AR1,AR1,AREA)
RAREA=FLOAT(AREA)
RAD=SQRT(RAREA/4.)
C COMPUTE FEATURE VECTOR
DO 10 I=1,12
C COPY IMAGE INTO AR2
CALL RESET (AR2)
CALL CHANGE (AR2,1,AR1,1,AR1,1)
C GENERATE DISPLACEMENTS AND SHIFT
XDISP=IFIX(RAD*COSINE(I))
YDISP=IFIX(RAD*SINE(I))
CALL SHIFT (AR2,XDISP,YDISP)
C COUNT INTERSECTIONS AND COMPUTE FEATURE I
CALL INTRSC (AR1,AR2,NI)
FTR(I)=IFIX(100.*FLOAT(NI)/RAREA)

```

```

IF (ISENS=0),EQ,0) GO TO 10
CALL CHANGE (AR2,1,AR1,1,AR1,1)
CALL ERASE
CALL DISPLAY (AR2)
CONTINUE
RETURN
END

```

10

```

C ARRAY PROCESSING ROUTINE XTRCT1
C G. KAUFMAN 2/74
C THIS ROUTINE EXTRACTS THE 0 DEG. EDGE SHAPE OF THE
C INPUT 360 X 240 BINARY IMAGE.
C
SUBROUTINE XTRCT1 (AR1,AREA,ANG)
REAL DIST,COSINE(12),SINE(12)
INTEGER ANG,AR1(20,240),AR2(20,240),AREA,I1,I2,I3,I4,I5,
1 XDISP,YDISP
COMMON /SCRATCH/AR2
DATA COSINE(1),COSINE(2),COSINE(3),COSINE(4),COSINE(5),
1 COSINE(6),COSINE(7),COSINE(8),COSINE(9),COSINE(10),
2 COSINE(11),COSINE(12)/1.,.96593,.86603,.70711,.5,.25882,
3 0.,.25882,.5,.70711,.86603,.96593/
DATA SINE(1),SINE(2),SINE(3),SINE(4),SINE(5),SINE(6),
1 SINE(7),SINE(8),SINE(9),SINE(10),SINE(11),SINE(12)
2 /0.,.25882,.5,.70711,.86603,.96593,1.,-.96593,
3 -.86603,-.70711,-.5,-.25882/
C COMPUTE ANGLE AND DISTANCE
DIST=SQRT(FLOAT(AREA))/3.
I1=ANG/15+1
C COPY IMAGE
CALL RESET (AR2)
CALL CHANGE (AR2,1,AR1,1,AR1,1)

```

```

C  SELECT SHIFT DIRECTION
DO 10 I2=1,5
I4=-1
IF ((I2.NE.1).AND.(I2.NE.5)) GO TO 20
I3=11
GO TO 40
IF (I2.NE.2) GO TO 30
I3=11+6
IF (I3.GT.12) I3=I3-12
GO TO 40
IF (I2.NE.3) GO TO 40
I4=2
C  COMPUTE SHIFT PARAMETERS
40  XDISP=FIX(DIST*COSINE(I3))*I4
    YDISP=FIX(DIST*SINE(I3))*I4
C  SHIFT AND EXTRACT
DO 50 I5=1,5
CALL SHIFT (AR2,XDISP,YDISP)
CALL CHANGE (AR1,0,AR2,1,AR2,1)
IF (I2.NE.5) GO TO 10
CONTINUE
CONTINUE
RETURN
END
50
10

```

```

.TITLE INTRSC
/  ARRAY PROCESSING ROUTINE INTRSC
/  G. KAUFMAN 1/74
/  THIS ROUTINE COUNTS THE NUMBER OF CELLS THAT
/  ARE '1' IN BOTH INPUT ARRAYS.
/
.GLOBAL INTRSC,.DA
/INCREMENT - NO SKIP INTENDED
INC=ISZ

```

```

INTRSC 0 JMS*
        JMP      .+4
        0
        0
        0
        LAW
        TAD
        DAC
        LAC*
        CMA
        TAD
        DAC
        LAC*
        DAC
        LAC*
        DAC
        DZM
        LAW
        DAC
        LAC*
        AND*
        RAL
        SZL
        INC
        ISZ
        JMP
        INC
        INC
        ISZ
        JMP
        LAC
        DAC*
        JMP*
        .END

        ARRAY1
        ARRAY2
        NIAD

        -3
        ARRAY1
        COUNT2*
        COUNT2
        01
        COUNT2
        ARRAY1
        ARRAY1
        ARRAY2
        ARRAY2
        01#
        -22
        COUNT1#
        ARRAY1
        ARRAY2

        /COMPUTE ARRAY SIZE ADDRESS
        /GET ARRAY SIZE
        /COMPUTE LOOP2 INDEX

        /GET ARRAY1 ADDRESS
        /GET ARRAY2 ADDRESS

        /INITIALIZE INTERSECTION COUNTER
        /INITIALIZE LOOP2 COUNTER

        /GET INTERSECTIONS

        /LOOP1 COUNTS INTERSECTIONS PER WORD
        /SKIP ON NO INTERSECTION
        /INTERSECTION - INCREMENT COUNTER
        /END OF WORD?
        /NO - LOOP
        /YES - GET NEXT WORD

        /END OF ARRAYS?
        /NO - LOOP
        /YES - PUT NI IN PARAMETER

        NIAD
        INTRSC
        /AND RETURN

```

```

      .TITLE  INVERT
      /  ARRAY PROCESSING ROUTINE INVERT
      /  G. KAUFMAN      1/74
      /  THIS ROUTINE INVERTS THE INPUT ARRAY.
      /
      .CLOBL  INVERT,,DA
      /INCREMENT - NO SKIP INTENDED
      INC=IS2
      INVERT
      0
      JMS*    .DA
      JMP     .+2
      ARRAY  0
      LAW     -3
      TAD     ARRAY
      QAC     COUNT#
      LAC*    COUNT
      CMA     (1
      TAD     COUNT
      JAC     ARRAY
      LAC*    ARRAY
      DAC     ARRAY
      LAC*    ARRAY
      CMA
      DAC*    ARRAY
      INC     ARRAY
      IS2     COUNT
      JMP     LOOP1
      JMP*    INVERT
      .END

      LOOP1
      /GET ARRAY SIZE
      /COMPUTE LOOP INDEX
      /GET ARRAY ADDRESS
      /GET ARRAY ELEMENT
      /INVERT
      /AND REPLACE
      /GO TO NEXT ELEMENT
      /END OF ARRAY?
      /NO - CONTINUE
      /YES - RETURN

```



```

.TITLE SHIFT
/  ARRAY PROCESSING ROUTINE SHIFT
/  G. KAUFMAN      1/74
/  THIS ROUTINE TRANSLATES THE INPUT ARRAY.
/
.GLOBAL SHIFT,ASIZE,.DA
INC=ISZ
SHIFT 0
      JMS* .DA
      JMP .+4
      0
      0
      0
      0
LAW
TAD
DAC
LAC*
TAD*
DAC
JMS*
JMP
.DSA
.DSA
.DSA
LAC*
DAC
LAC
TAD
DAC
LAC
CMA
TAD
DAC
TAD
DAC
XOIST
YDIST
      -3
      ARRAY
      SIZE#
      SIZE
      ARRAY
      SIZE
      ASIZE
      .+4
      ARRAY+400000
      NCOLS#
      /AND STORE
      NCOLS#
      /GET ARRAY START ADDRESS
      NCOLS
      NCOLS2=2*NCOLS
      NCOLS
      NCOLS#
      /NCOLSN=-NCOLS
      (1
      NCOLSN#
      NCOLSN
      /NCOLN2=-2*NCOLS
      NCOLN2#
      /COMPUTE ARRAY SIZE ADDRESS
      /GET ARRAY SIZE
      /FINAL ADDRESS + 1
      /AND STORE
      /GET ARRAY DIM.
      1
      1

```

HORIZ	LAC	STRT	/INITIALIZE ARRAY
	DAC	ARRAY	/ADDRESS
	LAC*	XOIST	/RIGHT OR LEFT ?
	SMA		
LEFT	JMP	RIGHT	
	CMA		/GET ABS(OIST)
	TAD	(1	/DIVIDE BY 18
	LMO		
	CLA:CLL		
	DIV		
	22		
	TAD	LSHFT	/REMAINDER GIVES
	DAC	SHFT1	/SHIFT LENGTH
	LACQ		/QUOTIENT+ARRAY ADDRESS
	TAD	ARRAY	/GIVES START ADDRESS
	DAC	FROM#	
	TAD	(1	
	DAC	FROM1#	/START ADDRESS+1
	LAC	ARRAY	/COMPUTE END OF ROW
	TAD	NCOLS	
	CMA		/AND TAKE NEG.
	TAD	(1	
	DAC	END#	
	LAC	FROM	/GET FIRST WORD ADDRESS
	DAC	FRM#	
	LAC	FROM1	/AND SECOND
	DAC	FRM1#	
	CLL		
LOOP1	LAC*	FRM1	/GET SECOND WORD
	LMO		
	LAC*	FRM	/AND FIRST
	XX		/SHIFT
SHFT1	DAC*	ARRAY	/AND DEPOSIT
	INC	FROM	/BUMP POINTERS
	INC	FROM1	
	INC	ARRAY	

LAC	ARRAY	
TAD	E.O.	
SPA		/NEXT ROW?
JMP	CONT1	/NO
LAC	NCOLSN	/YES
TAD	END	/BUMP END OF
DAC	END	/ROW POINTER
TAD	SIZE	
SMA		/END OF ARRAY?
JMP	CONT1	/NO
JMP	VERT	/YES - GO TO VERT
LAC	FROM1	
TAD	END	
SPA		/IN THIS ROW?
JMP	CONT2	/YES - GET NEXT WORD
LAC	ZR	/NO - GET ZERO
JMP	CONT3	
LAC	FROM1	
DAC	FRM1	
LAC	FROM	
TAD	END	
SPA		/IN THIS ROW?
JMP	CONT4	/YES - GET NEXT WORD
LAC	ZR	/NO - GET ZERO
JMP	CONT5	
LAC	FROM	
DAC	FRM	
JMP	LOOP1	
LLS		/THEN LOOP
ZER		/LEFT SHIFT INSTRUCTION
ZER		/ZERO ADDRESS POINTERS
0		
LMQ		/DIVIDE DIST BY 18
CLA:CLL		
DIV		
22		
CMA		/18-REMAINDER
81TS		

TAD	(1		
TAD	BITS		
TAD	LSHIFT		/GIVES SHIFT LENGTH
DAC	SHIFT2		
LACQ			/ARRAY ADDRESS+NCOLS
CMA			/-1-QUOTIENT
TAD	ARRAY		/G VES START ADDRESS
TAD	NCOLS		
DAC	FROM		/START ADDRESS-1
TAD	(-1		
DAC	FROM1		/COMPUTE END OF ROW
LAC	ARRAY		/AND TAKE NEG.
CMA			
TAD	(1		
DAC	END		/COMPUTE DEPOSIT ADDRESS
LAC	ARRAY		
TAD	(-1		
TAD	NCOLS		
DAC	ARRAY		
LAC	FROM		/GET FIRST WORD ADDRESS
DAC	FRM		
LAC	FROM1		/AND SECOND
DAC	FRM1		
CLL			
LAC*	FRM		/GET FIRST WORD
LMQ			
LAC*	FRM1		/AND SECOND
XX			/SHIFT
DAC*	ARRAY		/AND DEPOSIT
LAC	ARRAY		/DECR POINTER
TAD	(-1		
DAC	ARRAY		
TAD	END		
SMA			/NEXT ROW?
JMP	CONT6		/NO

LOOP2

SHIFT2


```

CON14  DAC
      JMP
      LAC
      DAC
      LAC*
      SMA
      JMP
      CMA
      TAD
      DAC
      LAC
      CLL
      MUL
      XX
      LACQ
      CMA
      TAD
      DAC
      LAC
      TAD
      DAC
      TAD
      DAC
      LAC
      CMA
      TAD
      DAC
      LAC
      DAC
      LAC*
      DAC*
      INC
      INC
      LAC
      TAD

      /THEN LOOP
      /INITIALIZE ARRAY
      /ADDRESS
      /UP OR DOWN?

      UP
      /GET ABS(DIST)

      (1
      MTPL1
      NCOLS
      /-DIST*NCOLS=FROM

      (1
      FROM
      NCOLSN
      SIZE
      FROM1
      FROM
      FROM
      SIZE
      /GIVES DEPOSIT ADDRESS
      /DEPOSIT ADDRESS-DIST*NCOLS
      /=FROM ADDRESS
      /COMPUTE END OF LINE

      (1
      END
      FROM
      FROM
      FROM1
      FROM
      FROM1
      FROM1
      END
      /GET FIRST WORD ADDRESS
      /GET WORD
      /AND DEPOSIT
      /BUMP POINTERS

```

LOOP3


```

TAD      (1
DAC      END
LAC      FROM
DAC      FRM
LAC*     FRM
DAC*     ARRAY
INC      FROM
INC      ARRAY
LAC      ARRAY
TAD      END
SMA      /END OF ARRAY?
JMP      /YES - RETURN
LAC      /NO
TAD      /OUT OF ARRAY?
SMA      /YES - GET ZERO
JMP      /NO - GET NEXT WORD
LAC      CON17
JMP      FROM
LAC      CON18
JMP      ZR
DAC      FRM
JMP      LOOP4
CLA      SHIFT
JMP*     .END

LOOP4
CON17
CON18
RTRN

```

```

/GET FIRST WORD ADDRESS

```

```

/GET WORD
/AND DEPOSIT
/BUMP POINTERS

```

```

/END OF ARRAY?
/YES - RETURN
/NO

```

```

/OUT OF ARRAY?
/YES - GET ZERO
/NO - GET NEXT WORD

```


REFERENCES

1. Fischer, G. L., Pollock, D. K., Raddack, B., and Stevens, M. E., Optical Character Recognition, Spartan Books, Washington, D. C., 1962.
2. Kirsch, R. A., Cahn, L., and Urban, G. H., "Experiments in Processing Pictorial Information with a Digital Computer," Proc. Eastern Joint Computer Conference, pp. 221-229, 1957.
3. Marill, T., and Green, D. M., "Statistical Recognition Functions and the Design of Pattern Recognizers," IRE Trans. on Electronic Computers, Vol. EC-9, No. 4, pp. 472-477, December 1960.
4. Chow, C. K., "An Optimum Character Recognition System Using Decision Functions," IRE Trans. on Electronic Computers, Vol. EC-6, No. 4, pp. 247-254, December 1957.
5. Dudani, S. A., An Experimental Study of Moment Methods for Automatic Identification of Three-Dimensional Objects from Television Images, Ph.D. dissertation, The Ohio State University, Columbus, Ohio, August 1973.
6. Carl, J. W., and Hall, C. F., "The Application of Filtered Transforms to the General Classification Problem," IEEE Trans. on Computers, Col. C-21, pp. 785-793, July 1972.
7. Richard, Jr., C. W., and Hemami, H., "Identification of Three-Dimensional Objects Using Fourier Descriptors of the Boundary Curve," IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-4, No. 4, pp. 371-378, July 1974.
8. Goldstein, R. J., Harmon, L. D., Lesk, A. B., "Identification of Human Faces," Proc. of the IEEE, Vol. 59, No. 5, pp. 748-760, May 1971.
9. Dudani, S. A., Breeding, K. J., and McGhee, R. G., "An Experimental System for Automatic Identification of Complex Three-Dimensional Objects from Television Images," submitted to IEEE Trans. on Computers, June 1974.
10. Goldstein, A. J., Harmon, L. D., Lesk, A. B., "Man-Machine Interaction in Human Face Identification," The Bell System Technical Journal, Vol. 51, No. 2, pp. 399-427, February 1972.
11. Bertillon, A., Signaletic Instructions, Werner Co., Chicago, 1893.

12. Duda, R. O., and Hart, P. E., Pattern Classification and Scene Analysis, John Wiley and Sons, New York, 1973.
13. Allen, A. L., Personal Descriptions, Butterworth and Co., London, 1950.
14. Harmon, L. D., "The Recognition of Faces," Scientific American, Vol. 229, No. 5, pp. 71-82, November 1973.
15. Kaya, Y., and Kobayashi, K., "A Basic Study on Human Face Recognition," Frontiers of Pattern Recognition, Academic Press, New York, pp. 265-289, 1971.
16. Goldstein, A. G., Mackenberg, E. J., "Recognition of Human Faces from Isolated Facial Features: A Developmental Study," Psychonomic Science, Vol. 6, No. 4, pp. 149-150, 1966.
17. Harmon, L. D., "Some Aspects of Recognition of Human Faces," in Pattern Recognition in Biological and Technical Systems, Springer-Verlag, New York, 1971.
18. Harmon, L. D., Julesz, B., "Masking in Visual Recognition: Effects of Two-Dimensional Filtered Noise," Science, Vol. 180, pp. 1194-1197, June 1973.
19. Hochberg, J., Galper, R. E., "Recognition of Faces: 1. An Exploratory Study," Psychonomic Science, Vol. 9, No. 12, pp. 619-620, 1967.
20. Bradshaw, J. L., Wallace, G., "Models for the Processing and Identification of Faces," Perception and Psychophysics, Vol. 9, No. 5, pp. 443-448, 1971.
21. Miller, G. A., "The Magic Number Seven Plus or Minus Two, or Some Limits on Our Capacity for Processing Information," Psychological Review, No. 63, pp. 81-97, 1956.
22. Gillenson, M., Generation of Facial Images, Ph.D. dissertation, The Ohio State University, Columbus, Ohio, June 1974.
23. Parke, F. I., "Computer Generated Animation of Faces," Proc. of the ACM Annual Conference, Vol. 1, pp. 451-457, 1972.
24. Wall, P. M., Eye-Witness Identification in Criminal Cases, Charles C. Thomas, Springfield, Illinois, 1965.
25. Willis, F., Faces and Features, Foster Art Service, Inc., Tustin, California,

26. Levine, M. D., "Feature Extraction: A Survey," Proc. of the IEEE, Vol. 50, No. 8, pp. 1391-1407, August 1973.
27. Unger, S. H., "A Computer Oriented Toward Spatial Problems," Proc. IRE, Vol. 46, pp. 1744-1750, October 1958.
28. Unger, S. H., "Pattern Detection and Recognition," Proc. IRE, Vol. 47, pp. 1737-1752, October 1959.
29. Golay, M. J., "Hexagonal Parallel Pattern Transformations," IEEE Trans. on Computers, Vol. C-18, pp. 733-740, August 1969.
30. Grey, S. B., "Local Properties of Binary Images in Two Dimensions," IEEE Trans. on Computers, Vol. C-20, pp. 551-561, May 1971.
31. Preston, K., "Feature Extraction by Golay Hexagonal Pattern Transformations," IEEE Trans. on Electronic Computers, Vol. EC-20, No. 9, pp. 1007-1014, September 1971.
32. Hawkins, J. K., Elerding, G. T., Bixby, K. W., and Haworth, P. A., "Automatic Shape Detection for Programmed Terrain Classification," Proc. Soc. Photographic Instrumentation Engrs., Boston, Mass., June 1966.
33. Nagy, G., "State of the Art in Pattern Recognition," Proc. of the IEEE, Vol. 56, pp. 836-862, May 1968.
34. Andrews, H. C., Computer Techniques in Image Processing, Academic Press, New York, 1970.
35. Fu, M. K., "Visual Pattern Recognition by Moment Invariants," IRE Trans. on Information Theory, Vol. IT-8, pp. 179-187, February 1962.
36. Patrick, E. A., Fundamentals of Pattern Recognition, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1972.
37. Cover, T. M., and Hart, P. E., "Nearest Neighbor Pattern Classification," IEEE Trans. on Infor. Theory, Vol. IT-13, January 1967.
38. Sebestyen, A. S., Decision-Making Processes in Pattern Recognition, The Macmillan Co., New York, 1962.
39. Miller, P. E., General Array Processing Package for the PDP-9, The Ohio State University, December 1973.

40. Yamada, H., and Amoroso, S., "Tessellation Automata," Information and Control, Vol. 13, No. 3, pp. 299-317, March 1969.
41. Miller, P. E., Picture Decomposition on a Parallel Processor, The Ohio State University, Columbus, Ohio, June 1973.
42. Horwitz, L. P., and Shelton, G. L., "Pattern Recognition Using Autocorrelation," Proc. IRE, Vol. 49, pp. 175-185, January 1961.
43. Rosenthal, A., Picture Processing by Computers, Academic Press, New York, 1969.
44. Duda, R. O., and Fossum, H., "Pattern Classification by Iteratively Determined Linear and Piecewise Linear Discriminant Functions," IEEE Trans. on Electronic Computers, Vol. EC-15, pp. 221-232, April 1966.
45. Jagadeesa, J. M., A Real Time Image Processing System Using a Color Television Camera, Ph.D. dissertation, The Ohio State University, Columbus, Ohio, August 1974.